INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.228

OPEN SYSTEMS INTERCONNECTION

PICS PROFORMAS

RELIABLE TRANSFER: PROTOCOL SPECIFICATION

ITU-T Recommendation X.228

(Extract from the *Blue Book*)

NOTES

1 ITU-T Recommendation X.228 was published in Fascicle VIII.5 of the *Blue Book*. This file is an extract from the *Blue Book*. While the presentation and layout of the text might be slightly different from the *Blue Book* version, the contents of the file are identical to the *Blue Book* version and copyright conditions remain unchanged (see below).

2 In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Recommendation X.228

RELIABLE TRANSFER: PROTOCOL SPECIFICATION[1)]

(Melbourne, 1988)

The CCITT,

considering

(a) that Recommendation X.200 defines the Basic Reference Model of Open Systems Interconnection (OSI) for CCITT applications;

(b) that Recommendation X.210 defines the service conventions for describing the services of the OSI reference model;

(c) that Recommendation X.216 defines the Presentation Layer service;

(d) that Recommendation X.217 defines the Association Control service;

(e) that Recommendation X.218 defines the Reliable Transfer service;

(f) that there is a need for common Reliable Transfer support for various applications;

unanimously declares

that this Recommendation defines the Reliable Transfer protocol of Open Systems Interconnection for CCITT Applications as given in the Scope and Field of Application.

CONTENTS

1) Recommendation X.228 and ISO 9066-2 [Information processing systems - Text Communication - ReliableTransfer Part 2: Protocol Specification] were developed in close collaboration and are technically aligned.

0 Introduction

This Recommendation specifies the protocol for the services provided by an application-service- element - the Reliable Transfer Service Element (RTSE) - to provide for the Reliable Transfer of Application Protocol Data Units (APDUs) between open systems. This Recommendation is one of a set of Recommendations specifying the protocols for sets of application-service-elements commonly used by a number of applications.

Reliable Transfer provides an application-independent mechanism to recover from communication and end-system failure minimizing the amount of retransmission.

This Recommendation is technically aligned with ISO 9066-2.

1 Scope and Field of Application

This Recommendation specifies the protocol (abstract syntax) and procedures for the Reliable Transfer Service Element services (Recommendation X.218). The RTSE services are provided in conjunction with the Association Control Service Element (ACSE) services (Recommendation X.217) and the ACSE protocol (Recommendation X.227), and the presentation-service (Recommendation X.216).

The RTSE procedures are defined in terms of:

a) the interactions between peer RTSE protocol machines through the use of ACSE and the presentation-service.

b) the interactions between the RTSE protocol machine and its service-user.

This Recommendation specifies conformance requirements for systems implementing these procedures.

2 References

Recommendation X.200 -	Reference Model of Open Systems Interconnection for CCITT Applications (See also ISO 7498)
Recommendation X.208 -	Specification of abstract syntax notation (See also ISO 8824).
Recommendation X.209 -	Specification of Basic Encoding Rules for the abstract syntax notation (See also ISO 8825).
Recommendation X.210 -	Open Systems Interconnection Layer Service Definition Conventions (See also ISO/TR 8509).
Recommendation X.216 -	Presentation Service Definition for Open Systems Interconnection for CCITT Applications (See also ISO 8822).
Recommendation X.217 -	Association Control Service Definition for CCITT Applications (See also ISO 8649).
Recommendation X.218 -	Reliable Transfer: Model and Service Definition (See also ISO 9066-1)
Recommendation X.219 -	Remote Operations: Model, Notation and Service Definition (See also ISO 9072-1)
Recommendation X.227 -	Association Control Protocol Specification for CCITT Applications (See also ISO 8650).

3 Definitions

3.1 *Reference Model Definitions*

This Recommendation is based on the concepts developed in Recommendation X.200 and makes use of the following terms defined in it:

a) Application Layer;

b) application-process;

c) application-entity;

d) application-service-element;

e) application-protocol-data-unit;

f) application-protocol-control-information;

g) presentation-service;
h) presentation-connection;
i) session-service;
j) session-connection; and
k) user-element
l) two-way-alternate interaction
m) transfer syntax.

3.2 *Service Conventions Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.210:

a) service-provider;
b) service-user;
c) confirmed service;
d) non-confirmed service;
e) provider-initiated service;
f) primitive;
g) request (primitive);
h) indication (primitive);
i) response (primitive); and
j) confirm (primitive).

3.3 *Presentation Service Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.216:

a) abstract syntax;
b) abstract syntax name;
c) presentation context;
d) default context.

3.4 *Association Control Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.217:

a) application-association; association;
b) application context;
c) Association Control Service Element; and
d) X.410-1984 mode.

3.5 *RTSE Service Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.218:

a) association-initiating-application-entity; association-initiator;
b) association-responding-application-entity; association-responder;
c) sending-application-entity; sender;
d) receiving-application-entity; receiver;
e) requestor;
f) acceptor;

g) Reliable Transfer Service Element;
h) RTSE-user;
i) RTSE-provider;
j) ACSE-provider;
k) monologue interaction;
l) syntax-matching-services;
m) Reliable Transfer;
n) X.410-1984 mode; and
o) normal mode.

3.6 *Reliable Transfer Protocol Specification Definitions*

For the purpose of this Recommendation the following definitions apply:

3.6.1 **reliable-transfer-protocol-machine:**

The protocol machine for the Reliable Transfer Service Element specified in this Recommendation.

3.6.2 **requesting-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the requestor of a particular Reliable Transfer Service Element service.

3.6.3 **accepting-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the acceptor for a particular Reliable Transfer Service Element service.

3.6.4 **sending-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the sender.

3.6.5 **receiving-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the receiver.

3.6.6 **association-initiating-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the association-initiator.

3.6.7 **association-responding-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the association-responder.

4 Abbreviations

4.1 *Data Units*

APDU application-protocol-data-unit.

4.2 *Types of Application-protocol-data-units*

The following abbreviations have been given to the application-protocol-data-units defined in this Recommendation.

RTAB	RT-P-ABORT and RT-U-ABORT application-protocol-data-unit
RTORQ	RT-OPEN-REQUEST application-protocol-data-unit
RTOAC	RT-OPEN-ACCEPT application-protocol-data-unit
RTORJ	RT-OPEN-REJECT application protocol-data-unit
RTTR	RT-TRANSFER application-protocol-data-unit
RTTP	RT-TOKEN-PLEASE application-protocol-data-unit

4.3 *Other Abbreviations*

The following abbreviations are used in this Recommendation.

AE	application-entity
ACSE	Association Control Service Element
ASE	application-service-element
RTPM	reliable-transfer-protocol-machine
RT (or RTS)	Reliable Transfer
RTSE	Reliable Transfer Service Element

5 Conventions

This Recommendation employs a tabular presentation of its APDU fields. In clause 7, tables are presented for each RTSE APDU. Each field is summarized using the following notation:

M	presence is mandatory
U	presence is a RTSE service-user option
T	presence is a RTPM option
req	source is related request primitive
ind	sink is related indication primitive
resp	source is related response primitive
conf	sink is related confirm primitive
sp	source or sink is the RTPM

The structure of each RTSE APDU is specified in clause 9 using the abstract syntax notation of Recommendation X.208.

6 Overview of the Protocol

6.1 *Service Provision*

The protocol specified in this Recommendation provides the services defined in Recommendation X.218. These services are listed in Table 1/X.228.

TABLE 1/X.228

RTSE Service Summary

Service	Type
RT-OPEN	Confirmed
RT-CLOSE	Confirmed
RT-TRANSFER	Confirmed
RT-TURN-PLEASE	Non-confirmed
RT-TURN-GIVE	Non-confirmed
RT-P-ABORT	Provider-initiated
RT-U-ABORT	Non-confirmed

6.2 *Use of Services*

6.2.1 *ACSE Services*

The RTPM requires access to the A-ASSOCIATE, A-RELEASE, A-ABORT and A-P-ABORT services. This Recommendation assumes that the RTPM is the sole user of these services.

6.2.2 *Use of the Presentation-service*

The RTPM requires access to the P-ACTIVITY-START, P-DATA, P-MINOR-SYNCHRONIZE, P-ACTIVITY-END, P-ACTIVITY-INTERRUPT, P-ACTIVITY-DISCARD, P-U-EXCEPTION-REPORT, P-ACTIVITY-RESUME, P-P-EXCEPTION-REPORT, P-TOKEN-PLEASE and P-CONTROL-GIVE services. This Recommendation assumes that the RTPM is the sole user of the above services.

The RTPM requires access to local syntax-matching-services provided by the presentation-service provider. This syntax-matching-service consists of:

a) an encoding service enabling the transformation from the local representation of an APDU value into an encoded-APDU-value of type OCTET STRING the value of which is the representation of the APDU value specified by the negotiated transfer syntax;

b) a decoding service enabling the transformation from an encoded-APDU-value into the local representation of the APDU value.

If X.410-1984 mode or simple encoding is used by the Presentation Layer, the APDU value is encoded as ASN.1 type ANY. If full encoding is used by the Presentation Layer, the APDU value is encoded as ASN.1 type EXTERNAL. (For X.410-1984 mode, single encoding and full encoding see Recommendation X.226).

This Recommendation recognizes that the ACSE services require access to the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services. This Recommendation assumes that the ACSE and the RTPM are the sole users of any of the above, or of any other, presentation-services.

During the lifetime of an application-association, the underlying presentation-connections either use a single presentation context or multiple presentation contexts as a part of the presentation multiple defined context facility. The choice is determined by the use of the Single Presentation Context parameter of the RT-OPEN service as described in clause 8.1.1.1.3 and 8.1.1.1.4.

6.3 *Model*

The reliable-transfer-protocol-machine (RTPM) communicates with its service-user by means of primitives defined in Recommendation X.218. Each invocation of the RTPM controls a single application-association.

The RTPM is driven by RTSE service request and response primitives from its service-user, and by indication and confirm primitives from the ACSE services and the presentation-service. The RTPM, in turn, issues indication and confirm primitives to its service-user and request and response primitives on the used ACSE services or presentation-service.

The reception of an RTSE service primitive, or of an ACSE service primitive, or of a presentation-service primitive, and the generation of dependent actions are considered to be indivisible.

During the use of the RTSE services, the existence of both the association-initiating AE and the association-responding AE is presumed. How these AEs are created is beyond the scope of this Recommendation.

During the use of the RTSE services, except RT-OPEN, the existence of an application-association between the peer AEs is presumed.

Note - Each application-association may be identified in an end system by an internal, implementation dependent mechanism so that the RTSE service-user, and the RTPM, and the ACSE service-provider can refer to it.

7 Elements of procedure

The RTSE protocol consists of the following elements of procedure:

a) association-establishment

b) association-release

c) transfer

d) turn-please

e) turn-give

f) error reporting:
   - f1) user-exception-report
   - f2) provider-exception-report

g) error handling:
   - g1) transfer-interrupt
   - g2) transfer-discard
   - g3) association-abort
   - g4) association-provider-abort

h) error recovery:
   - h1) transfer-resumption (for recovery from g1), or after successful h3) from g3) or g4))
   - h2) transfer-retry (for recovery from g2))
   - h3) association-recovery (for recovery from g3 or g4))

i) abort:
   - i1) transfer-abort (recovery from g1) or g2) or g3) or g4) not possible)
   - i2) provider-abort (recovery from g1) or g2) or g3) or g4) not possible)
   - i3) user-abort.

In the following clauses, a summary of each of these elements of procedure is presented. This consists of a summary of the relevant APDUs, and a high-level overview of the relationship between the RTSE service primitives and the APDUs involved and the used presentation-service.

Clause 8 describes how the service primitives are mapped on the ACSE services, and on the presentation-service.

7.1 *Association-establishment*

7.1.1 *Purpose*

The association-establishment procedure is used to establish an application-association.

7.1.2 *APDUs used*

The association-establishment procedures uses the RT-OPEN-REQUEST (RTORQ) APDU, the RT-OPEN-ACCEPT (RTCAC) APDU, and the RT-OPEN-REJECT (RTORJ) APDU.

Note - These APDUs are also used in the association-recovery procedure.

7.1.2.1 *RTORQ APDU*

The RT-OPEN REQUEST (RTORQ) APDU is used in the request to establish an application-association. The fields of the RTORQ APDU are listed in Table 2/X.228.

7.1.2.2 *RTOAC APDU*

The RT-OPEN-ACCEPT (RTOAC) APDU is used in the positive response to the request to establish an application-association. The fields of the RTOAC APDU are listed in Table 3/X.228.

TABLE 2/X.228

RTORQ APDU Fields

Field name	Presence	Source	Sink
Checkpoint-size	T	sp	sp
Window-size	T	sp	sp
Dialogue-mode	U	req	ind
User-data (Note 1)	U	req	ind
Session-connection-identifier (Note 2)	T	sp	sp
Application-protocol (Note 3)	U	req	ind

Note 1 - The User-data field is used solely in the association-establishment procedure.

Note 2 - The Session-connection-identifier field is used solely in the association-recovery procedure.

Note 3 - The Application-protocol field is used solely in the X.410-1984 mode.

TABLE 3/X.228

RTOAC APDU Fields

Field name	Presence	Source	Sink
Checkpoint-size	T	sp	sp
Window-size	T	sp	sp
User-data (Note 1)	U	resp	conf
Session-connection-identifier (Note 2)	T	sp	sp

Note 1 - The User-data field is used solely in the association-establishment procedure.

Note 2 - The Session-connection-identifier field is used solely in the association-recovery procedure.

7.1.2.3 *RTORJ APDU*

RT-OPEN-REJECT (RTORJ) APDU is used in the negative response to the request to establish an application-association. The fields of the RTORJ APDU are listed in Table 4/X.228.

TABLE 4/X.228

RTORJ APDU Fields

Field name	Presence	Source	Sink
Refuse-reason (Note 1)	T	sp	sp
User-data (Note 2)	U	resp	conf

Note I - The Refuse-reason field is used solely in the X.410-1984 mode.

Note 2 - The User-data field is used solely in the normal mode, and is not used in the association-recovery procedure.

7.1.3 *Association-establishment procedure*

This procedure is driven by the following events:

a) an RT-OPEN request primitive from the requestor (association-initiator);

b) an RTORQ APDU as user-data on an A-ASSOCIATE indication primitive;

c) an RT-OPEN response primitive from the acceptor (association-responder);

d) an A-ASSOCIATE confirm primitive that may contain either an RTOAC APDU, or an RTORJ APDU, or no APDU.

7.1.3.1 *RT-OPEN request primitive*

The requesting RTPM forms an RTORQ APDU from the parameter values of the RT-OPEN request primitive and its internal data. The RT-OPEN request primitive parameters, except user-data, are stored by the requesting RTPM for association-recovery. The requesting RTPM issues an A-ASSOCIATE request primitive also using information from the RT-OPEN request primitive. The RTORQ APDU is the user information parameter value of the A-ASSOCIATE request primitive.

The requesting RTPM waits for a primitive from the ACSE-provider and does not accept any other primitive from the requestor.

7.1.3.2 *RTORQ APDU*

If the application-association is not accepted by the ACSE-provider, no A-ASSOCIATE indication primitive is received by the accepting RTPM and no action takes place.

If the application-association is accepted by the ACSE-provider, the accepting RTPM receives the RTORQ APDU as the user information parameter on an A-ASSOCIATE indication primitive.

If any of the A-ASSOCIATE indication parameters or any of the RTORQ APDU fields is unacceptable to the accepting RTPM, or if the accepting RTPM is not able to accept the application-association, it forms and sends an RTORJ APDU with appropriate parameters from internal data. The accepting RTPM issues an A-ASSOCIATE response primitive. The RTORJ APDU is sent as the user information parameter of the A-ASSOCIATE response primitive. The application-association is not established. The accepting RTPM does not issue an RT-OPEN indication.

If the A-ASSOCIATE indication primitive and the RTORQ APDU parameters are acceptable to the accepting RTPM, it issues an RT-OPEN indication primitive to the acceptor. The RT-OPEN indication parameter values are derived from the RTORQ APDU and the A-ASSOCIATE indication primitive parameter values.

The accepting RTPM waits for an RT-OPEN response primitive from the acceptor, or a primitive from the ACSE-provider.

7.1.3.3 *RT-OPEN response primitive*

When the accepting RTPM receives an RT-OPEN response primitive from the acceptor, the result parameter specifies whether the acceptor has accepted (value "accepted") or rejected the application-association.

If the application-association is accepted by the acceptor, the accepting RTPM forms an RTOAC APDU using the RT-OPEN response primitive parameters and internal data. The RT-OPEN response primitive parameters, except user data, are stored by the accepting RTPM for association-recovery.

The accepting RTPM issues an A-ASSOCIATE response primitive also using information from the RT-OPEN response primitive. The RTOAC APDU is sent as the user information parameter of the A-ASSOCIATE response primitive.

If the application-association is rejected by the acceptor, the accepting RTPM forms a RTORJ APDU using the RT-OPEN response primitive parameters and internal data. The accepting RTPM issues an A-ASSOCIATE response primitive also using information from the RT-OPEN request primitive. The RTORJ APDU is sent as the user information parameter of the A-ASSOCIATE response primitive. The application-association is not established.

7.1.3.4 *A-ASSOCIATE confirm primitive*

The requesting RTPM receives an A-ASSOCIATE confirm primitive. The following situations are possible:

a) the application-association has been accepted by the acceptor;

b) the accepting RTPM or the acceptor has rejected the application-association; or

c) the ACSE service provider has rejected the application-association.

If the application-association was accepted by the acceptor, the A-ASSOCIATE confirm primitive result parameter has the value "accepted" and the RTOAC APDU is the value of the user information parameter of the A-ASSOCIATE confirm primitive. The requesting RTPM issues an RT-OPEN confirm primitive to the requestor. The result parameter has the value "accepted" and the user-data parameter contains the user-data parameter value of the RTOAC APDU. The other parameters of the RT-OPEN confirm primitive are derived from the A-ASSOCIATE confirm primitive.

If the application-association was rejected by either the acceptor or the accepting RTPM, the A-ASSOCIATE confirm primitive result parameter has one of the values "rejected..", the A-ASSOCIATE confirm primitive result source parameter has the value "ACSE service-user" and the RTORJ APDU is the value of the user information parameter of the A-ASSOCIATE confirm primitive. The requesting RTPM issues an RT-OPEN confirm primitive to the requestor. The result parameter has one of the values "rejected . . ." and the other parameter values are derived from the A-ASSOCIATE confirm primitive parameters and the RTORJ APDU. The application-association is not established.

If the application-association was rejected by the ACSE service-provider, the A-ASSOCIATE confirm primitive result parameter has one of the values "rejected ... ", the A-ASSOCIATE confirm primitive result source parameter has either the value "ACSE service-provider" or "presentation service-provider". The user-data parameter of the RT-OPEN confirm primitive is absent and the application-association is not established. The other parameters of the RT-OPEN confirm primitive are derived from the A-ASSOCIATE confirm primitive.

7.1.4 *Use of the RTORQ APDU fields*

The RTORQ APDU fields are used as follows.

7.1.4.1 *Checkpoint-size*

The checkpoint-size field allows negotiation of the maximum amount of data (in units of 1024 octets) that may be sent between two minor synchronization points. A value of zero from the requesting RTPM invites the accepting RTPM to select checkpoint-size. If this field is absent, checkpoint-size zero is assumed.

7.1.4.2 *Window-size*

The window-size field allows negotiation of the maximum number of outstanding minor synchronization points before data transfer shall be suspended. If this field is absent, window-size 3 is assumed.

7.1.4.3 *Dialogue-mode*

This is the dialogue-mode parameter value from the RT-OPEN request primitive. It appears as the dialogue-mode parameter value of the RT-OPEN indication primitive.

The value of this field is either monologue, or two-way-alternate. If this field is absent, monologue is assumed.

7.1.4.4 *User-data*

This is the user-data parameter value from the RT-OPEN request primitive. It appears as the user-data parameter value of the RT-OPEN indication primitive.

The value of this field is transparent to the RTPM.

7.1.4.5 *Session-connection-identifier*

This field is used only in the association-recovery procedure.

7.1.4.6 *Application-protocol*

This field is used solely in the X.410-1984 mode. It is the application-protocol parameter value from the RT-OPEN request primitive. It appears as the application-protocol parameter value in the RT-OPEN indication primitive.

7.1.5 *Use of the RTOAC APDU fields*

The RTOAC APDU fields are used as follows.

7.1.5.1 *Checkpoint-size*

The checkpoint-size field allows negotiation of the maximum amount of data (in units of 1024 octets) that may be sent between two minor synchronization points. If the checkpoint-size in the RTORQ APDU is greater than zero, the accepting RTPM shall supply a value in the RTOAC APDU that is less than or equal to the value in the RTORQ APDU, else the accepting RPTM may select checkpoint-size. A value of zero from the accepting RTPM indicates that checkpointing will not be used. The value of this field becomes the agreed maximum value and governs both directions of transfer. If this field is absent, it is assumed that checkpointing will not be used.

7.1.5.2 *Window-size*

This field is only used if checkpoint-size of the RTOAC APDU is greater than zero. The window-size field allows negotiation of the maximum number of outstanding minor synchronization points before data transfer shall be suspended. The accepting RTPM shall supply a value that is less than or equal to the value in the RTORQ APDU. This becomes the agreed maximum size and governs both directions of transfer. If this field is absent, window-size 3 is assumed.

7.1.5.3 *User-data*

This is the user-data parameter value from the RT-OPEN response primitive. It appears as the user-data parameter value of the RT-OPEN confirm service primitive.

The value of this field is transparent to the RTPM.

7.1.5.4 *Session-connection-identifier*

This field is used only in the association-recovery procedure.

7.1.6 *Use of the RTORJ APDU fields*

The RTORJ APDU fields are used as follows.

7.1.6.1 *Refuse-reason*

The refuse-reason field is only used in the X.410-1984 mode.

This field may contain one of the following values:

- rts-busy	The accepting RTPM, or the acceptor, is so loaded that it cannot support a new application-association. The requesting RTPM should retry after a period of time. This value is either provided by the accepting RTPM, or is derived from the result parameter value "rejected (transient)" of the RT-OPEN response primitive from the acceptor. It appears as the result parameter value "rejected (transient)" of the RT-OPEN confirm primitive to the requestor.
- cannot recover	This value is used only by the accepting RTPM in the association-recovery procedure if it is unable to accept an association-recovery.
- validation failure	The acceptor does not recognize the requestor's credentials as being valid for the proposed application-association. This value is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.
- unacceptable-dialogue-mode	The acceptor does not accept the type of dialogue mode proposed for the application-association. This value is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.

7.1.6.2 *User-data*

This field is only used in the normal mode:

This is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.

The value of this field is transparent to the RTPM.

7.2 *Association-release*

7.2.1 *Purpose*

The association-release procedure is used for the normal release of an application-association by the association-initiator without loss of information in transit.

7.2.2 *APDUs used*

No APDUs are used in this procedure.

7.2.3 *Association-release procedure*

This procedure is driven by the following events:

a) an RT-CLOSE request primitive from the requestor (association-initiator);

b) an A-RELEASE indication primitive;

c) an RT-CLOSE response primitive from the acceptor (association-responder);

d) an A-RELEASE confirm primitive.

7.2.3.1 *RT-CLOSE request primitive*

The requestor may issue an RT-CLOSE request primitive only if it possesses the turn and if there is no outstanding RT-TRANSFER confirm primitive. When an RT-CLOSE request primitive is received from the requestor, the requesting (association-initiating) RTPM issues an A-RELEASE request primitive. The reason parameter of the A-RELEASE request primitive is the reason parameter of the RT-CLOSE request primitive. The user information parameter of the A-RELEASE request primitive is the user-data parameter of the RT-CLOSE request primitive.

Note - No RT-CLOSE request primitive parameters are present in X.410-1984 mode.

The requesting RTPM waits for a primitive from the ACSE service-provider and does not accept any other primitive from the requestor.

7.2.3.2 *A-RELEASE indication primitive*

The accepting RTPM receives the A-RELEASE indication primitive.

It issues an RT-CLOSE indication primitive to the acceptor. The RT-CLOSE indication parameter values are derived from the A-RELEASE indication primitive.

Note - No RT-CLOSE indication primitive parameters are present in X.410-1984 mode.

The RTPM waits for a primitive from the acceptor or the used service provider.

7.2.3.3 *RT-CLOSE response primitive*

When the accepting RTPM receives an RT-CLOSE response primitive, the accepting RTPM issues an A-RELEASE response primitive. The reason parameter of the A-RELEASE response primitive is the reason parameter of the RT-CLOSE response primitive. The user information parameter of the A-RELEASE response primitive is the user-data parameter of the RT-CLOSE response primitive. The result parameter value of the A-RELEASE response primitive is "affirmative".

Note - No RT-CLOSE response primitive parameters are present in X.410-1984 mode.

7.2.3.4 *A-RELEASE confirm primitive*

The requesting RTPM receives an A-RELEASE confirm primitive.

The requesting RTPM issues an RT-OPEN confirm primitive to the acceptor. The RT-OPEN confirm primitive parameter values are derived from the A-RELEASE confirm primitive.

Note - No RT-CLOSE confirm primitive parameters are present in X.410-1984 mode.

7.3 *Transfer*

7.3.1 *Purpose*

The transfer procedure is used to transfer an RTSE-user APDU from the requestor (sender) to the acceptor (receiver).

7.3.2 *APDUs used*

Each RTSE-user APDU, conveyed in an RT-TRANSFER request, constitutes an activity. For each application-association, at most one activity, or one interrupted activity awaiting resumption, may exist at a time.

The RTSE-user APDU value is transformed into the encoded-APDU-value and vice versa by means of the local syntax-matching services. The transfer procedure uses the RT-TRANSFER (RTTR) APDU. The transfer procedure supports segmenting and reassembling of the encoded-APDU-value into/from one or more RT-FR APDUs.

An encoded-APDU-value is transferred as a single RTTR APDU if checkpointing is not used. Otherwise, the encoded-APDU-value is transferred as a series of RTTR APDUs, the maximum size (i.e. number of octets forming the RTTR APDU value) of each being the negotiated checkpoint-size. The concatenation of the RTTR APDU values is the encoded-APDU-value.

The fields of the RTTR APDU are listed in Table 5/X.228.

TABLE 5/X.228

RTTR APDU Fields

Field name	Presence	Source	Sink
User-data-part	M	req	ind/conf

7.3.3 *Transfer procedure*

This procedure is driven by the following events:

a) an RT-TRANSFER request primitive from the requestor (sender);

b) a P-ACTIVITY-START indication primitive, followed by one or more RTTR APDUs as user-data of P-DATA indication primitives each, except the last, followed by a P-MINOR-SYNCHRONIZE indication primitive;

c) a P-MINOR-SYNCHRONIZE confirm primitive;

d) a P-ACTIVITY-END indication primitive;

e) a P-ACTIVITY-END confirm primitive;

f) a transfer time-out.

7.3.3.1 *RT-TRANSFER request primitive*

If the requesting RTPM possesses the turn and receives a RT-TRANSFER request from the requestor, the requesting RTPM transforms the RTSE-user APDU value into the encoded-APDU-value by means of the encoding service of the local syntax-matching services.

The requesting RTPM issues a P-ACTIVITY-START request primitive and may start transmitting the first RTTR APDU in a P-DATA request primitive immediately after the P-ACTIVITY-START request primitive is issued, since the latter service is not a confirmed service.

The maximum RTTR APDU size will have been negotiated during the association-establishment procedure. The requesting RTPM shall submit, in P-DATA request primitives, RTTR APDUs that conform to that agreement. Checkpoints may only be inserted if a checkpoint-size greater than zero was negotiated during the association-establishment procedure.

If a transferred RTTR APDU is not the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the requesting RTPM inserts a checkpoint by issuing a P-MINOR-SYNCHRONIZE request primitive. The requesting RTPM uses only the "explicit confirmation expected" type of minor synchronization. The requesting RTPM may issue further P-DATA request primitives and P-MINOR-SYNCHRONIZE request primitives unless the agreed window-size has been reached.

If the RTTR APDU is the only one, or the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the requesting RTPM issues a P-ACTIVITY-END request primitive.

Consecutive P-DATA request primitives shall not be issued, and all data transfer shall take place within an activity.

7.3.3.2 *P-ACTIVITY-START indication primitive, RTTR APDUs, and P-MINOR-SYNCHRONIZE indication primitives*

The accepting RTPM receives a P-ACTIVITY-START indication primitive, indicating the start of transfer of a RTSE-user APDU. The accepting RTPM receives an RTTR APDU as user-data of a P-DATA indication primitive.

If the RTTR APDU is not the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the accepting RTPM receives a P-MINOR-SYNCHRONIZE indication primitive. If the accepting RTPM has secured the RTTR APDU, it issues a P-MINOR-SYNCHRONIZE response primitive.

7.3.3.3 *P-MINOR-SYNCHRONIZE confirmed primitive*

When the requesting RTPM receives a P-MINOR-SYNCHRONIZE confirm primitive, it assumes that the accepting RTPM has secured the encoded-APDU-value APDU up to that point.

The requesting RTPM may issue further P-DATA request primitives and P-MINOR-SYNCHRONIZE request primitives unless the agreed window-size has been reached. The window is advanced when a P-MINOR-SYNCHRONIZE confirm primitive is received by the requesting RTPM.

When a complete encoded-APDU-value has been transmitted, the requesting RTPM issues a P-ACTIVITY-END request primitive.

7.3.3.4 *P-ACTIVITY-END indication primitive*

A P-ACTIVITY-END indication primitive indicates to the accepting RTPM that a complete encoded- APDU-value has been transferred. The accepting RTPM transforms the encoded-APDU-value into the RTSE-user APDU value by means of the decoding service of the local syntax-matching-services.

If the accepting RTPM has secured the complete RTSE-user APDU, it issues an RT-TRANSFER indication primitive to the acceptor, and issues a P-ACTIVITY-END response primitive.

The accepting RTPM records the session-connection-identifier and the activity identifier of the last RTSE-user APDU which it completely secured for association-recovery purposes.

7.3.3.5 *P-ACTIVITY-END confirm primitive*

An activity end is an implicit major synchronization point and once successfully confirmed by means of an P-ACTIVITY-END confirm primitive, it indicates to the requesting RTPM that the RTSE-user APDU has been secured by the accepting RTPM. The requesting RTPM may then delete the transferred RTSE-user APDU.

When the requesting RTPM receives the P-ACTIVITY-END confirm primitive, it issues an RT-TRANSFER confirm primitive with a result parameter value of "APDU-transferred" to the requestor.

7.3.3.6 *Transfer time-out*

If an APDU has not been transferred within the time specified in the transfer-time parameter of the RT-TRANSFER request primitive (that is, the requesting RTPM has not received the P-ACTIVITY-END confirm primitive), the requesting RTPM performs the transfer-discard procedure followed by the transfer-abort procedure.

If during the transfer-discard procedure, the requesting RTPM does not receive a P-ACTIVITY-DISCARD confirm primitive within a (locally specified) reasonable time, the requesting RTPM performs the transfer-abort procedure followed by the provider-abort procedure.

7.4 *Turn-please*

7.4.1 *Purpose*

The turn-please procedure is used by a receiver (requestor) to request the turn from the sender (acceptor).

7.4.2 *APDUs used*

The turn-please procedure uses the RT-TURN-PLEASE (RTTP) APDU.

The fields of the RTTP APDU are listed in Table 6/X.228.

TABLE 6/X.228

RTTP APDU Fields

Field name	Presence	Source	Sink
Priority	U	req	ind

7.4.3 *Turn-please procedure*

This procedure is driven by the following events:

a) an RT-TURN-PLEASE request primitive from the requestor;

b) an RTTP APDU as user-data of a P-TOKEN-PLEASE indication primitive.

7.4.3.1 *RT-TURN-PLEASE request primitive*

If the requesting RTPM does not possess the turn and receives an RT-TURN-PLEASE request from the requestor, the requesting RTPM issues a P-TOKEN-PLEASE request primitive. If the priority parameter is present in the RT-TURN-PLEASE request primitive, and RTTP APDU is formed from the parameter value and transferred as user-data of the P-TOKEN-PLEASE request primitive. This procedure may be performed either inside or outside an activity.

7.4.3.2 *RTTP APDU*

If the accepting RTPM receives a P-TOKEN-PLEASE indication primitive, the accepting RTPM issues an RT-TURN-PLEASE indication primitive to the acceptor. If an RTTP APDU is transferred as user-data of the P-TOKEN-PLEASE indication primitive, the RT-TURN-PLEASE indication primitive parameter is present and derived from the RTTP APDU.

7.4.4 *Use of the RTTP fields*

The RTTP APDU fields are used as follows.

7.4.4.1 *Priority*

This is the priority parameter value of the RT-TURN-PLEASE request primitive. It appears as the priority parameter value of the RT-TURN-PLEASE indication primitive.

The value of this field is transparent to the RTPM.

7.5 *Turn-give*

7.5.1 *Purpose*

The turn-give procedure is used by a sender (requestor) to give the turn to the receiver (acceptor). The requestor becomes the receiver and the acceptor becomes the sender.

7.5.2 *APDUs used*

No APDUs are used in this procedure.

7.5.3 *Turn-give procedure*

The turn-give procedure is driven by the following events:

a) an RT-TURN-GIVE request primitive;

b) a P-CONTROL-GIVE indication primitive.

7.5.3.1 *RT-TURN-GIVE request primitive*

If the requesting RTPM possesses the turn and receives an RT-TURN-GIVE request primitive from the requestor, it issues a P-CONTROL-GIVE request primitive and becomes the receiving RTPM. This may be done only outside an activity.

7.5.3.2 *P-CONTROL-GIVE indication primitive*

If the accepting RTPM receives a P-CONTROL-GIVE indication primitive, the accepting RTPM issues an RT-TURN-GIVE indication primitive to the acceptor, and issues a P-CONTROL-GIVE response primitive. The accepting RTPM becomes the sending RTPM.

7.6 *Error reporting*

7.6.1 *User-exception-report*

7.6.1.1 *Purpose*

The user-exception-report procedure is used by the receiving RTPM to report an error situation to the sending RTPM.

7.6.1.2 *APDUs used*

No APDUs are used in this procedure.

7.6.1.3 *User-exception-report procedure*

This procedure is driven by the following events:

a) a receiving RTPM problem;

b) a P-U-EXCEPTION-REPORT indication primitive.

7.6.1.3.1 *Receiving RTPM problem*

If the receiving RTPM detects a problem, it issues a P-U-EXCEPTION-REPORT request primitive and starts a local recovery timer. Depending on the severity of the detected error, the value of the reason parameter of the P-U-EXCEPTION-REPORT request primitive is as follows:

a) In severe problem situations, the value "receiving ability jeopardized" is used.

b) In exceptional circumstances, the receiving RTPM may have to delete a partially received RTSE-user APDU, even though some minor synchronization points have been confirmed. In this case, the value "unrecoverable procedure error" is used.

c) If the receiving RTPM is not willing to complete a transfer procedure, the value "non-specific error" is used.

d) If the sending RTPM resumes a transfer procedure already finished by the receiving RTPM (see clause 7.8.1.3.2), the value "sequence error" is used.

e) For all other less severe error situations, the value "local SS-user error" is used.

7.6.1.3.2 *P-U-EXCEPTION-REPORT indication primitive*

If the sending RTPM receives a P-U-EXCEPTION-REPORT indication primitive, it performs one of the following procedures depending on the reason parameter value of the P-U-EXCEPTION-REPORT indication primitive:

a) With a value "receiving ability jeopardized", the transfer-abort procedure followed by the provider abort procedure are performed.

b) With a value "unrecoverable procedure error", the transfer-discard procedure followed by transfer-retry procedure are performed.

c) With a value "non-specific error", the transfer-discard procedure followed by the transfer-abort procedure are performed.

d) With a value "sequence error", the transfer-discard procedure is performed and the requesting RTPM issues an RT-TRANSFER confirm primitive with a result parameter value of "APDU-transferred" to the requestor and the transfer procedure is finished.

e) With a value "local SS-user error" and at least one confirmed checkpoint in the transfer procedure, the transfer-interrupt procedure followed by the transfer-resumption procedure are performed. If no checkpoint was confirmed in the transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure are performed.

7.6.2 *Provider-exception-report*

7.6.2.1 *Purpose*

If the presentation service-provider detects an unexpected situation during an activity, not covered by other services, a P-P-EXCEPTION-REPORT indication primitive is issued to both RTPMs.

7.6.2.2 *APDUs used*

No APDUs are used in this procedure.

7.6.2.3 *Provider-exception-report procedure*

This procedure is driven by the following events:

a) a P-P-EXCEPTION-REPORT indication primitive.

7.6.2.3.1 *P-P-EXCEPTION-REPORT indication primitive*

The receiving RTPM receives a P-P-EXCEPTION-REPORT indication primitive.

If the sending RTPM receives a P-P-EXCEPTION-REPORT indication primitive, it may perform one of the following procedures:

a) if at least one checkpoint was confirmed in the transfer procedure, the transfer-interrupt procedure followed by the transfer-resumption procedure; or,

b) if no checkpoint was confirmed in the transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure; or,

c) the transfer-abort procedure followed by the provider-abort procedure.

7.7 *Error handling*

7.7.1 *Transfer-interrupt*

7.7.1.1 *Purpose*

The transfer-interrupt procedure is used by the sending RTPM to handle a less severe (than those handled by the other error handling procedures) error situation during the transfer procedure, if at least one checkpoint was confirmed during the transfer procedure.

7.7.1.2 *APDUs used*

No APDUs are used in this procedure.

7.7.1.3 *Transfer-interrupt procedure*

This procedure is driven by the following events:

a) a sending RTPM problem;

b) a P-ACTIVITY-INTERRUPT indication primitive;

c) a P-ACTIVITY-INTERRUPT confirm primitive.

7.7.1.3.1 *Sending RTPM problem*

If the sending RTPM detects a less severe problem and at least one checkpoint was confirmed during the transfer procedure, it issues a P-ACTIVITY-INTERRUPT request primitive with one of the following reason parameter values:

a) "non-specific error", if the problem was indicated by an error reporting procedure;

b) "local SS-user error", if the problem is a local sending RTPM problem.

7.7.1.3.2 *P-ACTIVITY-INTERRUPT indication primitive*

If the receiving RTPM receives a P-ACTIVITY-INTERRUPT indication primitive, it issues a P-ACTIVITY-INTERRUPT response primitive and starts a local recovery timer.

7.7.1.3.3 *P-ACTIVITY-INTERRUPT confirm primitive*

If the sending RTPM receives a P-ACTIVITY-INTERRUPT confirm primitive, it starts the transfer-resumption procedure.

7.7.2 *Transfer-discard*

7.7.2.1 *Purpose*

The transfer-discard procedure is used by the sending RTPM to escape from a more severe (than those handled by the transfer-interrupt procedure) error situation, or a less severe error situation if no checkpoint was confirmed, during the transfer procedure.

7.7.2.2 *APDUs used*

No APDUs are used in this procedure.

7.7.2.3 *Transfer-discard procedure*

This procedure is driven by the following events:

a) a sending RTPM problem;

b) a P-ACTIVITY-DISCARD indication primitive;

c) a P-ACTIVITY-DISCARD confirm primitive.

7.7.2.3.1 *Sending RTPM problem*

If the sending RTPM detects a more severe problem, or a less severe problem if no checkpoint was confirmed during the transfer procedure, it issues a P-ACTIVITY-DISCARD request primitive with one of the following Reason parameter values:

a) "non-specific error", if the problem was indicated by an error reporting procedure;

b) "local SS-user error", or "unrecoverable procedural error", if the problem is a local sending RTPM problem.

7.7.2.3.2 *P-ACTIVITY-DISCARD indication primitive*

If the receiving RTPM receives a P-ACTIVITY-DISCARD indication primitive, it issues a P-ACTIVITY-DISCARD response primitive. The receiving RTPM deletes all knowledge and contents of the associated RTSE-user APDU so far received.

If the receiving RTPM has already issued an RT-TRANSFER indication primitive, it performs the association-abort procedure. The abort-reason field value of the RTAB APDU is "transfer-completed". In this case the sending RTPM ends the transfer procedure with a positive RT-TRANSFER confirm primitive and the association-recovery procedure is performed.

7.7.2.3.3 *P-ACTIVITY-DISCARD confirm primitive*

The receipt of a P-ACTIVITY-DISCARD confirm primitive by the sending RTPM signifies the completion of the transfer-discard procedure.

7.7.3 *Association-abort*

7.7.3.1 *Purpose*

The association-abort procedure is used by the RTPMs to handle the most severe error situations. This procedure can be performed between an RT-TRANSFER request primitive and its corresponding RT-TRANSFER confirm primitive.

7.7.3.2 *APDUs used*

The association-abort procedure uses the RT-ABORT (RTAB) APDU. The fields of the RTAB APDU are listed in Table 7/X.228.

Note - The RTAB APDU is also used by the provider-abort and the user-abort procedure.

TABLE 7/X.228

RTAB APDU Fields

Field name	Presence	Source	Sink
Abort-reason	T	sp	sp
Reflected-parameter	T	sp	sp
User-data	U	req	ind

7.7.3.3 *Association-abort procedure*

This procedure is driven by the following events:

a) an RTPM-abort;

b) an RTAB APDU.

7.7.3.3.1 *RTPM-abort*

Either the receiving or the sending RTPM transfers an RTAB APDU to its peer as user-data of an A-ABORT request primitive. If the RTPM is the association-initiating RTPM, it performs the association recovery procedure. If the RTPM is the association-responding RTPM, it awaits association-recovery. The receiving RTPM starts a local recovery timer.

After successful association recovery, the sending RTPM performs the transfer-resumption procedure.

7.7.3.3.2 *RTAB APDU*

Either the sending RTPM or the receiving RTPM may receive an RTAB APDU as user-data of an A-ABORT indication primitive. If the RTPM is the association-initiating RTPM, it performs the association-recovery procedure. If the RTPM is the association-responding RTPM, it awaits association recovery. The receiving RTPM starts a local recovery timer.

After successful association recovery, the sending RTPM performs the transfer-resumption procedure.

7.7.3.4 *Use of the RTAB APDU fields*

The RTAB APDU fields are used as follows:

7.7.3.4.1 *Abort-reason*

This field may contain one of the following values:

- local-system-problem	
- invalid-parameter	The invalid parameters are specified in the reflected-parameter field.
- unrecognized-activity	The sending RTPM shall perform the transfer-abort procedure optionally followed by the provider-abort procedure.
- temporary-problem	No attempt at association-recovery should be made for a period of time determined by a local rule.
- protocol-error	Of the RTPM.
- permanent-error	This value is used solely by the provider-abort procedure in normal mode.
- user-abort	This value is used solely by the user-abort procedure in normal mode.
- transfer-completed	The receiving RTPM could not discard an already completed transfer.

7.7.3.4.2 *Reflected-parameter*

The reflected-parameter field is a bit string that identifies which parameters are regarded as invalid parameters in the primitive received from the used service by the aborting RTMP before the association-abort. The order of the bits in the bit string is the same as the order of the parameters in the tables of service parameters in Recommendations X.217 and X.216 (i.e. bit 1 represents the first parameter, etc.).

7.7.3.4.3 *User-data*

This field is not used in the association-abort procedure.

7.7.4 *Association-provider-abort*

7.7.4.1 *Purpose*

The association-provider-abort procedure is used to handle an ACSE-provider, or a presentation service-provider abort.

7.7.4.2 *APDUs used*

No APDUs are used in this procedure.

7.7.4.3 *Association-provider-abort procedure*

This procedure is driven by the following event:

a) an AP-ABORT indication primitive.

7.7.4.3.1*AP-ABORT indication primitive*

An association-provider-abort is indicated to both RTPMs by an AP-ABORT indication primitive, and may occur at any time.

After such an event, the association-initiating RTPM starts the association-recovery procedure. Both RTPMs start a local recovery timer.

If the association-provider-abort procedure was performed during the transfer procedure, the sending RTPM starts the transfer-resumption procedure after the association-recovery procedure is successfully completed. If the association-recovery procedure was not successfully completed, the sending RTPM performs the transfer-error procedure, and the provider-abort procedure.

7.8 *Error recovery*

7.8.1 *Transfer-resumption*

7.8.1.1 *Purpose*

The transfer-resumption procedure is used by the sending RTPM to recover from:

a) an error situation handled by the transfer-interrupt procedure; or,

b) an error situation handled by the association-abort procedure during a transfer procedure. In this case the transfer-resumption procedure is performed after an association-recovery procedure is successfully performed. If no checkpoint was confirmed in the interrupted transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure are performed after transfer-resumption.

7.8.1.2 *APDUs used*

The transfer-resumption procedure uses the RTTR APDU (see clause 7.3.2).

7.8.1.3 *Transfer-resumption procedure*

This procedure is driven by the following events:

a) the resumption of an interrupted activity;

b) a P-ACTIVITY-RESUME indication primitive.

After these events, the transfer procedure is used to continue (see clause 7.3.3).

7.8.1.3.1 *Resumption of an interrupted activity*

The sending RTPM issues a P-ACTIVITY-RESUME request primitive with parameters that link the resumed Activity to the previously interrupted one.

After the sending RTPM has issued the P-ACTIVITY-RESUME request primitive and at least one checkpoint was confirmed in the interrupted transfer procedure, it continues the transfer procedure by issuing a P-DATA request primitive for the RTTR APDU following the last confirmed checkpoint. If no checkpoint was confirmed in the interrupted transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure are performed.

7.8.1.3.2 *P-ACTIVITY-RESUME indication primitive*

If the receiving RTPM receives a P-ACTIVITY-RESUME indication primitive, it checks the Old Activity Identifier and the Old Session Connection Identifier parameters of the P-ACTIVITY-RESUME indication primitive with the corresponding information (session-connection-identifier, and Activity Identifier) recorded for the last completely secured transfer (see clause 7.3.3.4).

If the information coincides, the receiving RTPM either:

a) responds correctly to the sending RTPM according to the transfer procedure, but discards the data it receives, and does not issue an RT-TRANSFER indication primitive; or,

b) performs the user-exception-report procedure with a Reason parameter value of "sequence error".

If the information does not coincide, and the old Activity Identifier and the old Session Connection Identifier parameters match the corresponding information of the previously interrupted activity, the transfer-resumption procedure continues as for the transfer procedure with a P-DATA indication primitive for the RTTR APDU following the last confirmed checkpoint.

If the receiving RTPM cannot resume the activity, the receiving RTPM performs the user-exception-report procedure or the association-abort procedure.

7.8.2 *Transfer-retry*

7.8.2.1 *Purpose*

The transfer-retry procedure is used by the sending RTPM to recover from an error situation handled by the transfer-discard procedure.

The completion of this procedure is as for the transfer procedure.

7.8.2.2 *APDUs used*

The transfer-retry procedure uses the RTTR APDU (see clause 7.3.2).

7.8.2.3 *Transfer-retry procedure*

The sending RTPM performs the transfer procedure (see clause 7.3.3). A new Activity Identifier parameter value is used in the P-ACTIVITY-START request primitive.

7.8.3 *Association-recovery*

7.8.3.1 *Purpose*

The association-recovery procedure is used by the association-initiating RTPM to recover from an error situation handled by the association-abort procedure or the association-provider-abort procedure.

7.8.3.2 *APDUs used*

The association-recovery procedure uses the RT-OPEN-REQUEST (RTORQ) APDU, the RT-OPEN- ACCEPT (RTOAC) APDU, and the RT-OPEN-REJECT (RTORJ) APDU.

7.8.3.2.1*RTORQ APDU*

The RT-OPEN-REQUEST (RTORQ) APDU is used in the request to recover an application-association. The fields of the RTORQ APDU are listed in clause 7.1.2.1.

The following rules apply:

a) the User-data field is not used;

b) the Session-connection-identifier field is mandatory.

7.8.3.2.2*RTOAC APDU*

The RT-OPEN-ACCEPT (RTOAC) APDU is used in the positive response to the request to recover an application-association. The fields of the RTOAC APDU are listed in clause 7.1.2.2.

The following rules apply:

a) the User-data field is not used;

b) the Session-connection-identifier field is mandatory.

7.8.3.2.3*RTORJ APDU*

The RT-OPEN-REJECT (RTORJ) APDU is used in the negative response to the request to recover an application-association. The fields of the RTORJ APDU are listed in clause 7.1.2.3.

The following rules apply:

a) the Refuse-reason field is used solely in the X.410-1984 mode;

b) the User-data field is not used.

7.8.3.3 *Association-recovery procedure*

This procedure is driven by the following events:

a) an A-ASSOCIATE request primitive by the association-initiating RTPM;

b) an RTORQ APDU as user-data on an A-ASSOCIATE indication primitive;

c) an A-ASSOCIATE confirm primitive that may contain either an RTOAC APDU, or an RTORJ, or no APDU.

7.8.3.3.1*A-ASSOCIATE request primitive*

The association-initiating RTPM forms an RTORQ APDU from its internal data. The association-initiating RTPM issues an A-ASSOCIATE request primitive using information stored during the association-establishment procedure (see clause 7.1.3.1). The RTORQ APDU is the User Information parameter value of the A-ASSOCIATE request primitive.

The association-initiating RTPM waits for a primitive from the ACSE service-provider.

7.8.3.3.2*RTORQ APDU*

If the application-association is not accepted by the ACSE service-provider, no A-ASSOCIATE indication primitive is received by the association-responding RTPM and no action takes place.

If the application-association is accepted by the ACSE service-provider, the association-responding RTPM receives the RTORQ APDU as the User Information parameter on an A-ASSOCIATE indication primitive.

If any of the A-ASSOCIATE indication primitive parameters, or any of the RTORQ APDU fields, is unacceptable to the association-responding RTPM, or if the association-responding RTPM is not able to accept the application-association, it forms and sends an RTORJ APDU with appropriate parameters from internal data. The association-responding RTPM issues an A-ASSOCIATE response primitive. The RTORJ APDU is sent as the User Information parameter of the A-ASSOCIATE response primitive. The application-association is not recovered.

If the A-ASSOCIATE indication primitive parameters, and the RTORQ APDU fields are acceptable to the association-responding RTPM, the association-responding RTPM forms an RTOAC APDU using internal data. The association-responding RTPM issues an A-ASSOCIATE response primitive. The RTOAC APDU is sent as the User Information parameter of the A-ASSOCIATE response primitive.

7.8.3.3.3*A-ASSOCIATE confirm primitive*

The association-initiating RTPM receives an A-ASSOCIATE confirm primitive. The following situations are possible:

a) the association-recovery has been accepted;

b) the accepting RTPM has rejected the association-recovery; or

c) the ACSE service provider has rejected the association-recovery.

If the association-recovery was accepted, the A-ASSOCIATE confirm primitive Result parameter has the value "accepted" and the RTOAC APDU is the value of the User Information Parameter of the A-ASSOCIATE confirm primitive. The application-association is recovered successfully, and if the association-abort occurred during the transfer procedure, the sending RTPM continues with the transfer-resumption procedure.

If the association-recovery was rejected by the responding RTPM, the A-ASSOCIATE confirm primitive Result parameter has one of the values "rejected..", the A-ASSOCIATE confirm primitive Result Source parameter has the value "ACSE service-user" and the RTORJ APDU is the value of the User Information Parameter of the A-ASSOCIATE confirm primitive. The application-association is not recovered.

If the association-recovery was rejected by the ACSE service-provider, the A-ASSOCIATE confirm primitive Result parameter has one of the values "rejected..." and the A-ASSOCIATE confirm primitive Result Source parameter has either the value "ACSE service-provider" or "presentation service-provider". The application-association is not recovered.

If the application-association was not recovered, the association-recovery procedure is performed again by the association-initiating RTPM after a time determined by a local rule:

a) if the Result parameter of the A-ASSOCIATE confirm primitive has the following value "rejected (transient)"; or

b) if, in X.410-1984 mode, the Refuse-reason field of the RTORJ APDU has the value "rts-busy".

In all other cases a provider-abort is performed as follows.

If the association-initiating RTPM is the sending RTPM, and the association-abort occurred during the transfer procedure, the sending RTPM performs the transfer-abort procedure. The association-initiating RTPM performs the provider-abort procedure.

If the association-responding RTPM detects a recovery-time-out, the following actions take place. If the association-responding RTPM is the sending RTPM, and the association-abort occurred during the transfer procedure, the sending RTPM performs the transfer-abort procedure. The association-responding RTPM performs the provider-abort procedure.

7.8.3.4 *Use of the RTORQ APDU fields*

The RTORQ APDU fields are used as follows.

7.8.3.4.1 *Checkpoint-size*

See clause 7.1.4.1.

7.8.3.4.2 *Window-size*

See clause 7.1.4.2.

7.8.3.4.3 *Dialogue-mode*

See clause 7.1.4.3.

7.8.3.4.4 *User-data*

This field is not used in the association-recovery procedure.

7.8.3.4.5 *Session-connection-identifier*

The Session-connection-identifier is used to specify the original session connection used in the association-establishment procedure. This is used in order to relate the new session-connection to the existing application-association.

7.8.3.5 *Use of the RTOAC APDU fields*

The RTOAC APDU fields are used as follows.

7.8.3.5.1 *Checkpoint-size*

See clause 7.1.5.1.

7.8.3.5.2 *Window-size*

See clause 7.1.5.2.

7.8.3.5.3 *User-data*

This field is used only in the association-recovery procedure.

7.8.3.5.4 *Session-connection-identifier*

The Session-connection-identifier is used to specify the original session connection used in the association-establishment procedure. This is used in order to relate the new session-connection to the existing application-association.

7.8.3.6 *Use of the RTORJ APDU fields*

The RTORJ APDU fields are used as follows.

7.8.3.6.1 *Refuse-reason*

The Refuse-reason field is used solely in the X.410-1984 mode.

This field may contain one of the following values:

- rts-busy	The association-responding RTPM is so loaded that it cannot support the application-association. The association-initiating RTPM should retry after a period of time. This value is provided by the association-responding RTPM.
- cannot recover	This value is used by the association-responding RTPM, if it is unable to accept an association-recovery.

7.8.3.6.4 *User-data*

This field is not used in the association-recovery procedure.

7.9 *Abort*

These procedures are performed when a successful recovery from one of the error handling procedures is not possible.

7.9.1 *Transfer-abort*

7.9.1.1 *Purpose*

The transfer-abort procedure is used by the sending RTPM if the transfer of an RTSE-user APDU is not possible.

7.9.1.2 *APDUs used*

No APDUs are used in this procedure.

7.9.1.3 *Transfer-abort procedure*

The sending RTPM issues an RT-TRANSFER confirm primitive with a Result parameter value "APDU- not-transferred". The APDU parameter value is the RTSE-user APDU not transferred.

7.9.2 *Provider-abort*

7.9.2.1 *Purpose*

The provider-abort procedure is used by the RTPMs, if recovery is not possible.

7.9.2.2 *APDUs used*

If an application-association exists, the provider-abort procedure uses the RT-ABORT (RTAB) APDU. The RTAB APDU is specified in clause 7.7.3.2.

7.9.2.3 *Provider-abort procedure*

This procedure is driven by the following events:

a) an RTPM-abort;

b) an RTAB APDU;

c) local recovery time-out.

7.9.2.3.1 *RTPM-abort*

If an application-association exists, either the receiving or the sending RTPM transfers an RTAB APDU to its peer as the User-data parameter of an A-ABORT request primitive. The RTPM issues a RT-P-ABORT indication primitive to its RTSE-user.

7.9.2.3.2 *RTAB APDU*

If the sending or the receiving RTPM receives an RTAB APDU as the User-data parameter of an A-ABORT indication primitive, it issues an RT-P-ABORT indication primitive to its RTSE-user.

7.9.2.3.3 *Recovery time-out*

If an application-association does not exist and a local recovery time-out occurs, the RTPM issues an RT-P-ABORT indication primitive to its RTSE-user.

7.9.2.4 *Use of the RTAB APDU fields*

The RTAB APDU fields are used as follows.

7.9.2.4.1 *Abort-reason*

The value of this field is "permanent-error".

7.9.2.4.2 *Reflected-parameter*

This field is not used.

7.9.2.4.3 *User-data*

This field is not used.

7.9.3 *User-abort*

7.9.3.1 *Purpose*

The user-abort procedure is used by the requestor to abort an application-association.

7.9.3.2 *APDUs used*

The user-abort procedure uses the RT-ABORT (RTAB) APDU. The RTAB APDU is specified in clause 7.7.3.2.

7.9.3.3 *User-abort procedure*

This procedure is driven by the following events:

a) an RT-U-ABORT request primitive from the requestor;

b) an RTAB APDU as User-data of an A-ABORT indication primitive.

7.9.3.3.1 *RT-U-ABORT request*

If the requesting RTPM receives an RT-U-ABORT request primitive from the requestor, an RTAB APDU is formed from the parameter value of the RT-U-ABORT request primitive and transferred as User-data of an A-ABORT request primitive.

7.9.3.3.2 *RTAB APDU*

If the accepting RTPM receives the RTAB APDU as User-data of an A-ABORT indication primitive, the accepting RTPM issues an RT-U-ABORT indication primitive to the acceptor. The RT-U-ABORT primitive parameter is derived from the RTAB APDU.

7.9.3.4 *Use of the RTAB APDU fields*

The RTAB APDU fields are used as follows.

7.9.3.4.1 *Abort-reason*

The value of this field is "user-error".

7.9.3.4.2 *Reflected-parameter*

This field is not used.

7.9.3.4.3 *User-data*

This is the User-data parameter value of the RT-U-ABORT request primitive. It appears as the User-data parameter value of the RT-U-ABORT indication primitive.

7.10 *Rules for extensibility*

In addition to the procedures stated above the following rule also applies when processing APDUs defined in this Recommendation:

- Ignore parameters which are not defined in this Recommendation for RTORQ, RTOAC, and RTORJ APDUs.

8 **Mapping to used services**

This clause defines how an RTPM transfers APDUs by means of:

a) the ACSE services, or

b) the presentation-service.

Clause 8.1 defines the mapping on the ACSE services, and clause 8.2 defines the mapping on the presentation-service.

Identification of the named abstract syntax in use is assumed for all RTSE services and is mapped onto used services, however this is a local matter and outside the scope of this Recommendation.

8.1 *Mapping on the ACSE services*

This clause defines how the ACSE service primitives described in Recommendation X.217 are used by the RTPM. Table 8/X.228 defines the mapping of the RTSE service primitives and APDUs to the ACSE service primitives.

TABLE 8/X.228

ACSE Mapping Overview

RTSE service	APDU	ACSE service
RT-OPEN request/indication	RTORQ	A-ASSOCIATE request/indication
RT-OPEN response/confirm	RTOAC	A-ASSOCIATE response/confirm
RT-OPEN response/confirm	RTORJ	A-ASSOCIATE response/confirm
RT-CLOSE response/indication	-	A-RELEASE request/indication
RT-CLOSE response/confirm	-	A-RELEASE response/confirm
association-abort	RTAB	A-ABORT request/indication
association-provider-abort	-	A-P-ABORT indication
RT-P-ABORT indication	RTAB	A-ABORT request/indication
RT- U-ABORT request/indication	RTAB	A-ABORT request/indication

Clause 8.1.1 defines the mapping onto ACSE in normal mode. Clause 8.1.2 defines the mapping onto ACSE in X.410-1984 mode.

8.1.1 *Mapping on the ACSE services in normal mode*

8.1.1.1 *Association-establishment procedure*

The association-establishment procedure takes place concurrently with the underlying ACSE association establishment.

8.1.1.1.1 *Directly mapped parameters*

The following parameters of the RT-OPEN service primitives are mapped directly onto the corresponding parameters of the A-ASSOCIATE service primitives:

a) Mode

b) Application Context Name

c) Calling AP Title

d) Calling AP Invocation-identifier

e) Calling AE Qualifier

f) Calling AE Invocation-identifier

g) Called AP Title

h) Called AP Invocation-identifier

i) Called AE Qualifier

j) Called AE Invocation-identifier

k) Responding AP Title

l) Responding AP Invocation-identifier

m) Responding AE Qualifier

n) Responding AE Invocation-identifier

o) Result Source

p) Diagnostic

q) Calling Presentation Address

r) Called Presentation Address

s) Responding Presentation Address

t) Presentation Context Definition List

u) Presentation Context Definition Result List

v) Default Presentation Context Name

w) Default Presentation Context Result.

8.1.1.1.2 *Parameters not used*

The following parameters of the A-ASSOCIATE service primitives are not used:

a) Presentation Requirements

b) Initial Synchronization Point Serial Number.

8.1.1.1.3 *Use of the other A-ASSOCIATE request and indication primitive parameters*

8.1.1.1.3.1 *User information*

For both the A-ASSOCIATE request and indication primitives, the User Information parameter is used to carry the RTORQ APDU.

8.1.1.1.3.2 *Quality of service*

The parameters "Extended Control" and "Optimized Dialogue Transfer" are set to "not required." The remaining parameters are set such that default values are used.

8.1.1.1.3.3 *Session requirements*

This parameter is set by the association-initiating RTPM to select the following functional units:

a) Half-duplex functional unit

b) Exceptions functional unit

c) Minor Synchronize functional unit

d) Activity Management functional unit.

8.1.1.1.3.4 *Initial assignment of tokens*

The association-initiating RTPM will always request that the data token be available for either monologue or two-way alternate interactions.

The association-initiating RTPM will specify which RTPM will initially hold the data token (minor synchronize token and major/activity token) upon successful completion of the session-connection phase, according to the initial-turn parameter of the RT-OPEN request primitive.

The association-initiating RTPM shall assign all of the tokens to the same RTPM. The application-association may be rejected if this rule is violated. At any particular point in time, the holder of the tokens is referred to as the sending RTPM, the other as the receiving RTPM.

8.1.1.1.3.5 *Session connection identifier*

The association-initiating RTPM will supply a Session Connection Identifier, which will be used to uniquely identify the session-connection. This identifier is formed of the following components: SS-User Reference, Common Reference, and, optionally, Additional Reference Information. The SS-User Reference is conveyed as the Calling SS-User Reference by the association-initiating RTPM. Common Reference and Additional Reference Information are conveyed in similarly named parameters of the P-CONNECT primitive.

Each component, when present, will contain a data element of the appropriate type from the following definitions:

```
CallingSSuserReference ::= CHOICE{      T61String
                                             - - solely in X410-1984 mode - -,
                                        OCTET STRING
                                             - - solely in normal mode - - }

CommonReference ::= UTCTime


AdditionalReferenceInformation :: = T61String
```

8.1.1.1.4*Use of the other A-ASSOCIATE response and confirm primitive parameters*

8.1.1.1.4.1 *User information*

Note - This parameter only has relevance if the application-association is accepted by the ACSE service-provider.

For both the A-ASSOCIATE response and confirm primitives, the User Information parameter is used to carry the RTOAC APDU, if the application-association is accepted; or the RTORJ APDU, if the application-association is rejected by either the association-responding RTPM, or the association-responder.

8.1.1.1.4.2 *Result*

For the A-ASSOCIATE response primitive the Result parameter is set by the association-responding RTPM as follows:

a) If the association-responding RTPM rejects the application-association, the value of this parameter is set to either "rejected (transient)" or "rejected (permanent)".

b) If the association-responding RTPM accepts the request, the value of this parameter is derived from the Result parameter of the RT-OPEN response primitive.

8.1.1.1.4.3 *Quality of service*

This parameter has the same value as in the A-ASSOCIATE request and indication primitives.

8.1.1.1.4.4 *Session requirements*

This parameter has the same value as in the A-ASSOCIATE request and indication primitives.

8.1.1.1.4.5 *Initial assignment of tokens*

This parameter is not used.

8.1.1.1.4.6 *Session connection identifier*

This parameter has the same value as in the A-ASSOCIATE indication primitive. The Calling SS-User Reference value of the A-ASSOCIATE indication primitive is returned as Called SS-User Reference by the association-responding RTPM.

8.1.1.2 *Association-release procedure*

The association-release procedure takes place concurrently with the underlying ACSE association release.

8.1.1.2.1*Directly mapped parameters*

The following parameters of the RT-CLOSE service primitives are mapped directly onto the corresponding parameters of the A-RELEASE service primitives:

a) Reason

b) User-data (on User Information)

8.1.1.2.2*Use of the other A-RELEASE response and confirm primitive parameters*

8.1.1.2.2.1 *Result*

The value of this parameter is "affirmative".

8.1.1.3 *Association-provider-abort*

8.1.1.3.1 *Use of the A-P-ABORT indication primitive parameters*

The use of the A-P-ABORT indication primitive parameters are defined in Recommendation X.217.

8.1.1.4 *Association-recovery procedure*

The association-recovery procedure takes place concurrently with the underlying ACSE association establishment.

8.1.1.4.1 *Parameters from RT-OPEN service*

The following parameters of the RT-OPEN service primitives are stored by the RTPMs, and mapped directly onto the corresponding parameters of the A-ASSOCIATE service primitives:

a) Mode
b) Application Context Name
c) Calling AP Title
d) Calling AP Invocation-identifier
e) Calling AE Qualifier
f) Calling AE Invocation-identifier
g) Called AP Title
h) Called AP Invocation-identifier
i) Called AE Qualifier
j) Called AE Invocation-identifier
k) Responding AP Title
l) Responding AP Invocation-identifier
m) Responding AE Qualifier
n) Responding AE Invocation-identifier
o) Calling Presentation Address
p) Called Presentation Address
q) Responding Presentation Address
r) Presentation Context Definition List
s) Presentation Context Definition Result List
t) Default Presentation Context Name
u) Default Presentation Context Result.

8.1.1.4.2 *Parameters not used*

The following parameters of the A-ASSOCIATE service primitives are not used:

a) Presentation Requirements
b) Initial Synchronization Point Serial Number.

8.1.1.4.3 *Parameters used as in the association-establishment procedure*

The following parameters of the A-ASSOCIATE service primitives are used as described for the association-establishment procedure (see clause 8.1.1.1):

a) User Information
b) Quality of service

c) Session Requirements

d) Session Connection Identifier.

8.1.1.4.4 *Use of the other A-ASSOCIATE request and indication primitive parameters*

8.1.1.4.4.1 *Initial assignment of tokens*

The following rules apply:

a) If the association-initiating RTPM has the Turn, it specifies the value "requestor side".

b) If the association-initiating RTPM does not have the Turn, but has issued a P-CONTROL-GIVE request primitive with no confirmation that the tokens were received, it specifies the value "acceptor side". (Receipt of data serves as confirmation that the tokens were received.)

c) If the association-initiating RTPM does not have the tokens and does not have a P-CONTROL-GIVE request primitive outstanding, it specifies the value "acceptor chooses".

8.1.1.4.5 *Use of the other A-ASSOCIATE response and confirm primitive parameters*

8.1.1.4.5.1 *Initial assignment of tokens*

If the value of this parameter in the A-ASSOCIATE indication primitive was "acceptor chooses", the association-responding RTPM will either keep (value "acceptor side") or return (value "requestor side") the tokens depending upon whether it had them before the session-connection was aborted.

8.1.1.4.5.2 *Result*

If the association-responding RTPM rejects the application-association, the value of this parameter is set to either "rejected (transient)" or "rejected (permanent)", else it is set to "accepted".

8.1.1.5 *Association-abort, provider-abort and user-abort procedures*

8.1.1.5.1 *Use of the A-ABORT request and indication primitive parameters*

8.1.1.5.1.1 *Abort source*

This parameter value is "requestor".

8.1.1.5.1.2 *User information*

This parameter value is the RTAB APDU.

8.1.2 *Mapping on the ACSE services in X.410-1984 Mode*

8.1.2.1 *Association-establishment procedure*

The association-establishment procedure takes place concurrently with the underlying ACSE association establishment.

8.1.2.1.1 *Directly mapped parameters*

The following parameters of the RT-OPEN service primitives are mapped directly onto the corresponding parameters of the A-ASSOCIATE service primitives:

a) Mode
b) Result Source
c) Diagnostic
d) Calling Presentation Address
e) Called Presentation Address
f) Responding Presentation Address

8.1.2.1.2 *Parameters not used*

The following parameters of the A-ASSOCIATE service primitives are not used:

a) Application Context Name
b) Calling AP Title
c) Calling AP Invocation-identifier
d) Calling AE Qualifier
e) Calling AE Invocation-identifier
f) Called AP Title
g) Called AP Invocation-identifier
h) Called AE Qualifier
i) Called AE Invocation-identifier
j) Responding AP Title
k) Responding AP Invocation-identifier
1) Responding AE Qualifier
m) Responding AE Invocation-identifier
n) Presentation Context Definition List
o) Presentation Context Definition Result List
p) Default Presentation Context Name
q) Default Presentation Context Result.

8.1.2.1.3 *Parameters used as in normal mode*

The following parameters of the A-ASSOCIATE service primitives are used as in the normal mode (see clause 8.1.1):

a) User Information
b) Result
c) Quality of Service
d) Session Requirements
e) Initial Assignment of Tokens
f) Session Connection Identifier.

8.1.2.2 *Association-release procedure*

The association-release procedure takes place concurrently with the underlying ACSE association release.

8.1.2.2.1 *Parameters not used*

The following parameters of the A-RELEASE service primitives are not used:

a) Reason

b) User Information

8.1.2.3 *Association-provider-abort procedure*

8.1.2.3.1 *Use of the A-P-ABORT indication primitive parameters*

The use of the A-P-ABORT indication primitive parameters are defined in Recommendation X.217.

8.1.2.4 *Association-recovery procedure*

The association-recovery procedure takes place concurrently with the underlying ACSE association establishment.

8.1.2.4.1 *Parameters from RT-OPEN service*

The following parameters of the RT-OPEN service primitives are stored by the RTPMs, and mapped directly onto the corresponding parameters of the A-ASSOCIATE service primitives:

a) Mode

b) Calling Presentation Address

c) Called Presentation Address

d) Responding Presentation Address.

8.1.2.4.2 *Parameters not used*

The following parameters of the A-ASSOCIATE service primitives are not used:

a) Application Context Name

b) Calling AP Title

c) Calling AP Invocation-identifier

d) Calling AE Qualifier

e) Calling AE Invocation-identifier

f) Called AP Title

g) Called AP Invocation-identifier

h) Called AE Qualifier

i) Called AE Invocation-identifier

j) Responding AP Title

k) Responding AP Invocation-identifier

1) Responding AE Qualifier

m) Responding AE Invocation-identifier

n) Presentation Context Definition List

o) Presentation Context Definition Result List

p) Default Presentation Context Name

q) Default Presentation Context Result

r) Presentation Requirements

s) Initial Synchronization Point Serial Number.

8.1.2.4.3 *Parameters used as in normal mode*

The following parameters of the A-ASSOCIATE service primitives are used as in the normal mode (see clause 8.1.1):

a) User Information

b) Result

c) Quality of service

d) Session Requirements

e) Initial Assignment of Tokens

f) Session Connection Identifier.

8.1.2.5 *Association-abort, provider-abort and user-abort procedures*

8.1.2.5.1 *Parameters not used*

The following parameters of the A-ABORT service primitives are not used:

a) Abort Source

8.1.2.5.2 *Parameters used as in normal mode*

The following parameters of the A-ASSOCIATE service primitives are used as in the normal mode (see clause 8.1.1):

a) User Information

8.2 *Mapping on the presentation services*

This clause defines how the presentation service primitives described in Recommendation X.216 are used by the RTPM. Table 9/X.228 defines the mapping of the RTSE service primitives and APDUs on the presentation service primitives.

This Clause defines the mapping onto presentation services in both normal mode and in X.410-1984 mode.

8.2.1 *Transfer procedure*

8.2.1.1 *Use of the P-ACTIVITY-START request and indication primitive parameters*

8.2.1.1.1 *Activity identifier*

The Activity Identifier identifies the activity by means of a serial number. The first activity started on the session-connection is assigned the number 1. Each successive activity for that direction of transfer is assigned the next number. Thus numbering is separate for each direction of transfer.

The property required of Activity Identifiers is that they should uniquely identify an activity during a reasonable time interval within a particular session-connection, so that duplicates can be detected in the face of error situations. These identifiers are allocated by numbering the activities during a session, starting with one for the first and incrementing for each successive activity, and to represent the number by a data element of type INTEGER, encoded according to Recommendation X.209. It is unnecessary for the receiving RTPM to make assumptions on the allocation method, only to be able to compare two identifiers for equality, octet by octet.

TABLE 9/X.228

Presentation Mapping Overview

RTSE service	APDU	Presentation-service
RT-TRANSFER req	-	P-ACTIVITY-START req/ind
	RTTR	P-DATA req/ind
	-	P -MINOR-SYNCHRONIZE req/ind/resp/conf
RT-TRANSFER ind/conf	-	P-ACTIVITY-END req/ind/resp/conf
RT-TURN-PLEASE req/ind	RTTP	P-TOKEN-PLEASE req/ind
RT-TURN-GIVE req/ind	-	P-CONTROL-GIVE req/ind
user-conception-report	-	P-U-EXCEPTION-REPORT req/ind
provider-exception -report	-	P-P-EXCEPTION-REPORT ind
transfer-interrupt	-	P-ACTIVITY-INTERRUPT req/ind/resp/conf
transfer-discard	-	P-ACTIVITY-DISCARD req/ind/resp/conf
transfer-resumption	-	P-ACTIVITY-RESUME req/ind

req	request
ind	indication
resp	response
conf	confirm

8.2.1.1.2 *User data*

This parameter is not used.

8.2.1.2 *Use of the P-DATA request and indication primitive parameters*

8.2.1.2.1 *User data*

The maximum User data size (number of octets of the RTTR APDU value) will have been negotiated during the association-establishment procedure. The sending RTPM shall submit User data that conforms to that agreement.

8.2.1.3 *Use of the P-MINOR-SYNCHRONIZE service parameters*

8.2.1.3.1 *Type*

The RTPM uses only the "explicit confirmation expected" type of minor synchronization.

8.2.1.3.2 *Synchronization point serial number*

The session service-provider allocates checkpoint serial numbers and passes them to the sending and receiving RTPMs to associate with the transmitted data.

8.2.1.3.3 *User data*

This parameter is not used.

8.2.1.4 *Use of the P-ACTIVITY-END service parameters*

8.2.1.4.1 *Synchronization point serial number*

The serial number of the implied major synchronization point is allocated by the session service-provider and passed up to both RTPMs.

8.2.1.4.2 *User data*

This parameter is not used.

8.2.2 *Turn-please procedure*

8.2.2.1 *Use of the P-TOKEN-PLEASE request and indication primitive parameters*

8.2.2.1.1 *Tokens*

The receiving RTPM will only request the data token. Since the tokens cannot be separated, the sending RTPM always surrenders all of the other available tokens when issuing the P-CONTROL-GIVE request primitive.

8.2.2.1.2 *User data*

This is the RTTP APDU.

8.2.3 *Turn-give procedure*

8.2.3.1 *Use of the P-CONTROL-GIVE service parameters*

The P-CONTROL-GIVE service primitives have no parameters. The data, minor synchronize, and major/activity tokens are automatically passed to the other RTPM.

8.2.4 *User-exception-report procedure*

8.2.4.1 *Use of the P-U-EXCEPTION-REPORT service parameters*

8.2.4.1.1 *Reason*

This parameter may specify one of the following reasons:

a) receiving ability jeopardized

b) local SS-User error

c) sequence error

d) unrecoverable procedure error

e) non-specific error.

8.2.4.1.2 *User data*

This parameter is not used.

8.2.5 *Provider-exception-report procedure*

8.2.5.1 *Use of the P-P-EXCEPTION-REPORT service parameters*

8.2.5.1.1 *Reason*

One of the following reason codes shall be supplied:

a) protocol error

b) non-specific error.

8.2.6 *Transfer-interrupt Procedure*

8.2.6.1 *Use of the P-ACTIVITY-INTERRUPT service parameters*

8.2.6.1.1 *Reason*

This parameter may specify one of the following:

a) local SS-User error

b) non-specific error.

8.2.7 *Transfer-discard procedure*

8.2.7.1 *Use of the P-ACTIVITY-DISCARD service parameters*

8.2.7.1.1 *Reason*

This parameter may specify one of the following:

a) local SS-User error

b) unrecoverable procedure error

c) non-specific error.

8.2.8 *Transfer-resumption procedure*

8.2.8.1 *Use of the P-ACTIVITY-RESUME service parameters*

8.2.8.1.1 *Activity identifier*

The sending RTPM shall allocate and supply the next Activity Identifier number for the current session.

8.2.8.1.2 *Old activity identifier*

The sending RTPM shall supply the original Activity Identifier which was assigned to the previously interrupted activity in the P-ACTIVITY-START request primitive.

8.2.8.1.3 *Synchronization point serial number*

The sending RTPM will specify the Serial Number of the last confirmed checkpoint in the interrupted activity. The session service-provider will also set the current session serial number to this value. If there was no previously confirmed checkpoint, the activity cannot be continued. The sending RTPM shall then send a P-ACTIVITY-RESUME request primitive (with the Synchronization Point Serial Number set to zero), followed by a P-ACTIVITY-DISCARD request primitive.

8.2.8.1.4 *Old session connection identifier*

The sending RTPM may supply the Session Connection Identifier of the Session Connection during which the Activity was started; it shall supply it if that session connection is not the current one. This Session Connection Identifier is conveyed in the Calling SS-User Reference, Common Reference, and, optionally,

Additional Reference Information components of this parameter. The Called SS-User Reference component is not used.

8.2.8.1.5 *User data*

This parameter is not used.

9 Abstract Syntax Definition of APDUs

The abstract syntax of each RTSE APDU is specified in this clause using the abstract syntax notation of Recommendation X.208, and is shown in Figure 1/X.228.

```
Reliable-Transfer-APDU {joint-iso-ccitt reliable-transfer (3) apdus (0)} DEFINITIONS ::=

BEGIN

EXPORTS rTSE, rTSE-abstract-syntax,

      RTORQapdu, RTOACapdu, RTORJapdu, RTABapdu; - - for use by Presentation Layer only

IMPORTS APPLICATION-SERVICE-ELEMENT FROM       Remote-Operations-Notation-extension
                                               {joint-iso-ccitt remote-operations(4)
                                               notation-extension(2)};

          rTSE-abstract-syntax OBJECT IDENTIFIER     ::={joint-iso-ccitt reliable-transfer(3) abstract-syntax(2)}
          rTSE APPLICATION-SERVICE-ELEMENT           ::={joint-iso-ccitt reliable-transfer(3) aseID (1)}

          RTSE-apdus ::=CHOICE{
                                   rtorq-apdu      [16] IMPLICIT  RTORQapdu,
                                   rtoac-apdu      [17] IMPLICIT  RTOACapdu,
                                   rtorj-apdu      [18] IMPLICIT  RTORJapdu,
                                   rttp-apdu                      RTTPapdu,
                                   rttr-apdu                      RTTRapdu,
                                   rtab-apdu       [22] IMPLICIT  RTABapdu}

          - - Tags [19], [20], [21] are used by the values of the UNBIND macro of the RO-notation of
          - - Recommendation X.219. Tags [0] to [15] inclusive are reserved for the
          - - use by the APDUs of ROSE (Recommendation X229). Any occurrence of
          - - ANY in this module shall be replaced by a single ASN. 1 type (if any) in an RTSE-user
          - - protocol specification. In addition any RTSE-user protocol sharing a single named
          - - abstract syntax with the RTSE protocol shall use distinct tags for the single
          - - presentation data values in the user data parameters of the RT-CLOSE (if any) and
          - - RT- TRANSFER services. These tags shall be distinct from the tag values [16], [17],
          - - [18] and [22] and from the ASN. 1 types INTEGER and OCTET STRING.
          - - Note - The above conditions are ensured, if the RTSE-user protocol specification uses the
          - - RO-notation of Recommendation X229.

          - - In X.410-1984 mode only the components of RTORQapdu, RTOACapdu, RTORJapdu
          - - and RTABapdu are used by the presentation layer. This has the effect that the following
          - - APDU types appear in the protocol in X.410-1984 mode instead of the alternative types
          - - of the RTSE-apdus type:
          - -                      RTORQapdu
          - -                      RTOACapdu
          - -                      RTORJapdu
          - -                      RTTPapdu
          - -                      RTTRapdu
          - -                      RTABapdu
- - RTSE Protocol continued
```

FIGURE 1/X.228 (Part 1 of 3)

Abstract Syntax Specification of RTSE Protocol

```
- - RTSE Protocol continued


RTORQapdu ::=                    SET{

    checkpointSize               [0] IMPLICIT INTEGER DEFAULT 0,
    windowSize                   [1] IMPLICIT INTEGER DEFAULT 3.
    dialogueMode                 [2] IMPLICIT INTEGER {monologue(0), twa(1)} DEFAULT monologue,
    connectionDataRQ             [3] ConnectionData,
    applicationProtocol          [4] IMPLICIT INTEGER OPTIONAL - -solely in X.410-1984 mode- - }


RTOACapdu ::=                    SET{

    checkpointSize               [0] IMPLICIT INTEGER DEFAULT 0,
    windowSize                   [1] IMPLICIT INTEGER DEFAULT 3,
    connectionDataAC             [2] ConnectionData}


RTORJapdu ::=                    SET{

    refuseReason                 [0] IMPLICIT RefuseReason OPTIONAL, - - only in X.410-1984 mode
    userDataRJ                   [1] ANY OPTIONAL - - RTSE user data, only in normal mode- -}


RTTPapdu :: = - - priority- -    INTEGER


RTTRapdu::=                      OCTET STRING


RTABapdu ::=                     SET{

    abortReason                  [0] IMPLICIT AbortReason OPTIONAL,
    reflectedParameter           [1] IMPLICIT BIT STRING OPTIONAL,
                                 - - 8 bits maximum, only if abortReason is invalidParameter
    userdataAB                   [2] ANY OPTIONAL - - only in normal mode and if abortReason
                                                - - is userError - -}

- - RTSE Protocol continued
```

FIGURE 1 /X.228 (Part 2 of 3)

Abstract Syntax Specification of RTSE Protocol

```
- - RTSE Protocol continued

ConnectionData ::=        CHOICE{

      open                        [0] ANY, - - RTSE user data
                                            - - this alternative is encoded as [0] IMPLICIT NULL
                                            - - in the case of absence of RTSE user data,

      recover                     [1] IMPLICIT SessionConnectionidentifier}

SessionConnectionIdentifier ::=  SEQUENCE{
                                 CallingSSuserReference,
                                 CommonReference,
                                 [0] IMPLICIT AdditionaIReferenceInformation OPTIONAL}

RefuseReason ::=                 INTEGER{
                                         rtsBusy(0),
                                         cannotRecover(1),
                                         validationFailure(2),
                                         unacceptableDialogueMode(3)}

CallingSSuserReference ::=       CHOICE{ T61String              - - solely in X.410-1984 - -,
                                           OCTET STRING         - - solely in normal mode - - }

CommonReference ::=                      UTCTime

AdditionalReferenceInformation ::=       T61String

AbortReason ::=                  INTEGER{
                                 localSystemProblem(0),
                                 invalidParameter(1), - - reflectedParameter supplied
                                 unrecognizedActivity(2),
                                 temporaryProblem(3),
                                 - - the RTSE cannot accept a session for a period of time- -
                                 protocolError(4), - - RTSE level protocol error- -
                                 permanentProblem(5), - -provider-abort solely in normal mode - -
                                 userError(6), - - user-abort solely in normal mode- -
                                 transferCompleted(7), - - activity can't be discarded- - }

END     - - of RTSE Protocol
```

FIGURE 1/X.228 (Part 3 of 3)

Abstract Syntax Specification of RTSE Protocol

10 Conformance

An implementation claiming conformance to this Recommendation shall comply with the requirements in clauses 10.1 through 10.3.

10.1 *Statement requirements*

An implementor shall state the following:

a) the application context for which conformance is claimed, including whether the system supports normal **mode** X.410-1984 mode or both.

10.2 *Static requirements*

The system shall:

a) conform to the abstract syntax definition of APDUs defined in clause 9.

10.3 *Dynamic requirements*

The system shall:

a) conform to the elements of procedure defined in clause 7,

b) conform to the mappings to the used services, for which conformance is claimed, as defined in clause 8.

ANNEX A

(to Recommendation X.228)

RTPM State Tables

This annex forms an integral Part of this Recommendation.

A.1 *General*

This annex defines a single Reliable Transfer Protocol Machine (RTPM) in terms of a state table. The state table shows the interrelationship between the state of an application-association, the incoming events that occur in the protocol, the actions taken, and, finally the resultant state of the application-association.

The RTPM state table does not constitute a formal definition of the RTPM. It is included to provide a more precise specification of the elements of procedure defined in clause 7.

This annex contains the following tables:

a) Table A-1/X.228 specifies the abbreviated name, source, and name/description of each incoming event. The sources are:
    1) RTSE-user (RTSE-user);
    2) peer RTPM (RTPM-peer);
    3) Association Control Service Element (ACSE);
    4) Presentation service-provider (PS-provider); and
    5) RTPM (RTPM).

b) Table A-2/X.228 specifies the abbreviated name of each state of the RTPM.

c) Table A-3/X.228 specifies the abbreviated name, target, and name/description of each outgoing event. The targets are:
    1) RTSE-user (RTSE-user);
    2) peer RTPM (RTPM-peer);
    3) Association Control Service Element (ACSE);
    4) Presentation service-provider (PS-provider); and
    5) RTPM (RTPM).

d) Table A-4/X.228 specifies the predicates.

e) Table A-5/X.228 specifies the specific actions.

f) Tables A-6/X.228 through A-16/X.228 including specifies the RTPM state table using the abbreviations of the above tables.

For some events the source and the target is the RTPM (internal event). If the RTPM issues an internal event as part of an action taken, the RTPM awaits that internal event in the resultant state.

A.2 *Conventions*

The intersection of an incoming event (row) and a state (column) forms a cell.

In the state table, a blank cell represents the combination of an incoming event and a state that is not defined for the RTPM (see § A.3.1). Some states await solely some incoming events from the source RTPM (internal events). These states are marked by * and no other incoming events are considered.

A non-blank cell represents an incoming event and a state that is defined for the RTPM. Such a cell contains one or more action lists. An action list may be either mandatory or conditional. If a cell contains a mandatory action list, it is the only action list in the cell.

A mandatory action list contains:

a) optionally one or more outgoing events,

b) optionally one or more specific actions, and

c) a resultant state.

A conditional action list contains:

a) a predicate expression comprising predicates and Boolean operators (¬ represents the Boolean NOT, & represents the Boolean AND), and

b) a mandatory action list (this mandatory list is used only if the predicate expression is true).

A local collision between an incoming event from the RTSE-user and the association-recovery procedure is modeled by deferring that event until completion of the association-recovery procedure.

A.3 *Actions to be taken by the RTPM*

The RTPM state table defines the action to be taken by the RTPM in terms of an optional outgoing event, optional specific actions, and the resultant state of the application-association.

A.3.1 *Invalid intersections*

Blank cells indicate an invalid intersection of an incoming event and state. If such an intersection occurs, one of the following actions is taken:

a) If the incoming event comes from the RTSE-user, or is an internal event, any action taken by the RTPM is a local matter.

b) If the incoming event is related to a received APDU, PS-provider, or ACSE; either the RTPM-issues an appropriate internal event, or the RTPM issues both an RT-PAind outgoing event (to its RTSE-user) and an RTAB outgoing event (to its peer RTPM).

A.3.2 *Valid intersections*

If the intersection of the state and incoming event is valid, one of the following actions is taken:

a) If the cell contains a mandatory action list, the RTPM takes the actions specified.

b) If a cell contains one or more conditional action lists, for each predicate expression that is true, the RTPM takes the actions specified. If none of the predicate expressions are true, the RTPM takes one of the actions defined in § A.3.1.

A.4 *Definition of variables and timers*

The following variables and timers are specified.

A.4.1 *Association-initiating RTPM*

This Boolean variable is set TRUE if the RTPM is the association-initiating RTPM (specific action [a1], else set FALSE (specific action [a2]).

This Boolean variable is tested in the predicate p11.

A.4.2 *Checkpoint-confirmed*

This Boolean variable is TRUE, if at least one checkpoint was confirmed during the transfer procedure. It is set FALSE at the beginning of the transfer procedure (specific actin [a30]). It is set TRUE, if a P-MINOR-SYNCHRONIZE confirm primitive is issued to the sending RTPM (specific action [a32]).

A.4.3 *Outstanding-minor-syncs*

This Integer variable indicates the number of outstanding checkpoint confirmations during the transfer procedure. It is set to zero at the beginning of the transfer procedure (specific action [a30] and [a33]). It is incremented by one, if a P-MINOR-SYNCHRONIZE request primitive is issued by the sending RTPM (specific action [a31]). It is decremented by 1, if a P-MINOR-SYNCHRONIZE confirm primitive is issued to the sending RTPM (specific action [a32]).

The value of this variable is compared with the value of the window-size field of the RTOAC APDU in the predicate p32. The value of this variable is compared with the value zero in predicate p33.

A.4.4 *Transfer timer Tr*

This timer is used to control the transfer time. It is set to the value of the transfer-time parameter of the RT-TRANSFER request primitive (specific action [a30]). It is reset if a RT-TRANSFER response primitive is issued by the sending RTPM (specific action [a35]).

In the case of time out the internal event tr-timeout occurs.

A.4.5 *Recovery timer rec*

This timer is used to control the recovery time. It is set to a locally specified value in the recovery case (specific action [a38]). It is reset after successful recovery (specific action [a39]).

In the case of time out the internal event rec-timeout occurs.

TABLE A-1/X.228 (Part 1 of 3)

Incoming Event List

Abbreviated name	Source	Name and description
RT-OPreq	RTSE-user	RT-OPEN request primitive
RT-OPres +	RTSE-user	RT-OPEN response primitive (Result = "accepted")
RT-OPres -	RTSE-user	RT-OPEN response primitive (Result = "rejected")
RT-CLreq	RTSE-user	RT-CLOSE request primitive
RT-CLres	RTSE-user	RT-CLOSE response primitive
RT-TRreq	RTSE-user	RT-TRANSFER request primitive
RT-TPreq	RTSE-user	RT-TURN-PLEASE request primitive
RT-TGreq	RTSE-user	RT-TURN-GIVE request primitive
RT-UAreq	RTSE-user	RT-U-ABORT request primitive
RTORQ	RTPM-peer	RTORQ APDU as user data of an A-ASSOCIATE indication primitive
RTOAC	RTPM-Peer	RTOAC APDU as user data of an A-ASSOCIATE confirm primitive
RTORJ	RTPM-peer	RTORJ APDU as user data of an A-ASSOCIATE confirm primitive
RTAB	RTPM-peer	RTAB APDU as user data of an A-ABORT indication primitive
RTTR	RTPM-peer	RTTR APDU as user data of a P-DATA indication primitive
RTTP	RTPM-peer	P-TOKEN-PLEASE indication primitive optionally with RTTP APDU as user data

TABLE A-1/X.228 (Part 2 of 3)

Incoming Event List

Abbreviated name	Source	Name and description
A-ASCcnf-	ACSE	A-ASSOCIATE confirm primitive (Result = "rejected") no RTORJ APDU
A-RELind	ACSE	A-RELEASE confirm primitive
A-RELcnf	ACSE	A-RELEASE confirm primitive
A-PABind	ACSE	A-P-ABORT indication primitive
P-ASind	PS-provider	P-ACTIVITY-START indication primitive
P-MSind	PS-provider	P-MINOR-SYNCHRONIZE indication primitive
P-MScnf	PS-provider	P-MINOR-SYNCHRONIZE confirm primitive
P-AEind	PS-provider	P-ACTIVITY-END indication primitive
P-AEcnf	PS-provider	P-ACTIVITY-END confirm primitive
P-CGind	PS-provider	P-CONTROL-GIVE indication primitive
P-UEind	PS-provider	P-U-EXCEPTION-REPORT indication primitive
P-PEind	PS-provider	P-P-EXCEPTION-REPORT indication primitive
P-AIind	PS-provider	P-ACTIVITY-INTERRUPT indication primitive
P-AIcnf	PS-provider	P-ACTIVITY-INTERRUPT confirm primitive
P-ADind	PS-provider	P-ACTIVITY-DISCARD indication primitive
P-ADcnf	PS-provider	P-ACTIVITY-DISCARD confirm primitive
P-ARind	PS-provider	P-ACTIVITY RESUME indication primitive

TABLE A-1/X.228 (Part 3 of 3)

Incoming Event List

Abbreviated name	Source	Name and description
a-ab	RTPM	associated aborted, recover
a-res	RTPM	activity resumption by the receiving RTPM
a-ret	RTPM	activity completed, discarded, or interrupted
ass-ab	RTPM	start of association-abort procedure
ass-rec	RTPM	start of association-recovery procedure
ass-rec-neg	RTPM	association-recovery unsuccessful
next	RTPM	transfer of RTTR APDU
p-ab	RTPM	start of provider-abort procedure
r-problem-1	RTPM	receiving RTPM problem
r-problem-2	RTPM	receiving RTPM problem more severe than r-problem-1
rec-timeout	RTPM	recovery time out
rt-ab	RTPM	RTAB received
s-problem-1	RTPM	sending RTPM problem
s-problem-2	RTPM	sending RTPM problem more severe than s-problem-1
s-problem-3	RTPM	sending RTPM problem more severe than s-problem-2
tr-discard	RTPM	start of transfer-discard procedure
tr-interr	RTPM	start of transfer-interrupt procedure
tr-p-ab	RTPM	start of procedures transfer-abort followed by provider-abort
tr-pos	RTPM	transfer successful completed
tr-res	RTPM	start of transfer-resumption of transfer
tr-timeout	RTPM	transfer time out
transfer	RTPM	start of transfer or transfer-retry procedures
u-exr	RTPM	start of user-exception-report procedure

TABLE A-2/X.228 (Part 1 of 2)

RTPM States

Abbreviated name	Name and description
STA0	idle, unassociated
STA01	awaiting RTOAC, RTORJ, or A-ASCcnf-
STA02	awaiting RT-OPres+, or RT-OPres-
STA11	associated; RTPM is association-initiating RTPM and sending RTPM
STA12	associated; RTPM is association-initiating RTPM and receiving RTPM
STA21	associated; RTPM is association-responding RTPM and sending RTPM
STA22	associated; RTPM is association-responding RTPM and receiving RTPM
STA30	transfer; sending RTPM
STA31	suspended transfer; sending RTPM
STA32	awaiting P-AEcnf; sending RTPM
STA321*	awaiting tr-pos; sending RTPM
STA34*	awaiting tr-discard to be followed by RT-TRcnf+; sending RTPM
STA341	awaiting P-ADcnf to be followed by RT-TRcnf+; sending RTPM
STA35*	awaiting tr-discard to be followed by RT-TRcnf-; sending RTPM
STA351	awaiting P-ADcnf to be followed by RT-TRcnf-; sending RTPM
STA36*	awaiting tr-discard to be followed by transfer-retry procedure; sending RTPM
STA361	awaiting P-ADcnf to be followed by transfer-retry procedure; sending RTPM
STA37*	awaiting tr-interr to be followed by transfer-retry procedure; sending RTPM
STA371	awaiting P-AIcnf; sending RTPM
STA372*	awaiting tr-res; sending RTPM
STA38*	awaiting ass-ab; sending RTPM
STA381*	awaiting a-ab; transfer sending RTPM
STA39*	awaiting rt-ab; transfer sending RTPM
STA40	awaiting RTTR; transfer receiving RTPM
STA400	awaiting RTTR; ignored transfer receiving RTPM

TABLE A-2/X.228 (Part 1 of 2)

RTPM States

Abbreviated name	Name and description
STA41	awaiting P-MSind or P-AEind; transfer receiving RTPM
STA410	awaiting P-MSind or P-AEind; ignored transfer receiving RTPM
STA42	awaiting recovery after u-exr event; transfer receiving RTPM
STA43*	awaiting a-ret; transfer receiving RTPM
STA44*	awaiting u-exr; transfer receiving RTPM
STA45*	awaiting a-res; transfer receiving RTPM
STA48*	awaiting ass-ab; transfer receiving RTPM
STA481*	awaiting a-ab; transfer receiving RTPM
STA49*	awaiting rt-ab; transfer receiving RTPM
STA51*	awaiting ass-rec or ass-rec-neg; association-recovery procedure outside activity
STA510	awaiting RTOAC or RTORJ; association-recovery procedure outside activity
STA52	awaiting RTORQ ; association-recovery procedure outside activity
STA53*	awaiting ass-rec or ass-rec-neg; association-recovery procedure sending RTPM
STA531	awaiting RTOAC or RTORJ; association-recovery procedure sending RTPM
STA532	awaiting RTORQ; association-recovery procedure sending RTPM
STA54*	awaiting ass-rec or ass-rec-neg; association-recovery procedure receiving RTPM
STA541	awaiting RTOAC or RTORJ; association-recovery procedure receiving RTPM
STA542	awaiting RTORQ; association-recovery procedure receiving RTPM
STA70*	awaiting abort; unassociated
STA71*	awaiting abort; associated
STA72*	awaiting rt-ab outside transfer
STA91	awaiting RT-CLres
STA92	awaiting A-RELcnf

TABLE A-3/X.228 (Part 1 of 3)

Outgoing Event List

Abbreviated name	Source	Name and description
RT-OPind	RTSE-user	RT-OPEN indication primitive
RT-OPcnf+	RTSE-user	RT-OPEN confirm primitive (Result = "accepted")
RT-OPcnf-	RTSE-user	RT-OPEN confirm primitive (Result = "rejected")
RT-CLind	RTSE-user	RT-CLOSE indication primitive
RT-CLcnf	RTSE-user	RT-CLOSE confirm primitive
RT-TRind	RTSE-user	RT-TRANSFER indication primitive
RT-TPind	RTSE-user	RT-TURN-PLEASE indication primitive
RT-TRcnf+	RTSE-user	RT-TRANSFER confirm primitive (Result = "APDU-transferred")
RT-TRcnf-	RTSE-user	RT-TRANSFER confirm positive (Result = "APDU-not-transferred")
RT-TGind	RTSE-user	RT-TURN-GIVE indication primitive
RT-UAind	RTSE-user	RT-U-ABORT indication primitive
RT-PAind	RTSE-user	RT-P-ABORT indication primitive
RTORQ	RTPM-peer	RTORQ APDU as user data of an A-ASSOCIATE request primitive
RTOAC	RTPM-peer	RTORQ APDU as user data of an A-ASSOCIATE response primitive
RTORJ	RTPM-peer	RTORJ APDU as user data of an A-ASSOCIATE response primitive
RTAB	RTPM-peer	RTAB APDU as user data of an A-ABORT request primitive
RTTR	RTPM-peer	RTTR APDU as user data of a P-DATA request primitive
RTTP	RTPM-peer	P-TOKEN-PLEASE indication primitive optionally with RTTP APDU as user data

TABLE A-3/X.228 (Part 2 of 3)

Outgoing Event List

Abbreviated name	Source	Name and description
A-RELreq	ACSE	A-RELEASE request primitive
A-RELres	ACSE	A-RELEASE response primitive
P-ASreq	PS-provider	P-ACTIVITY-START request primitive
P-MSreq	PS-provider	P-MINOR-SYNCHRONIZE request primitive
P-MSres	PS-provider	P-MINOR-SYNCHRONIZE response primitive
P-AEreq	PS-provider	P-ACTIVITY-END request primitive
P-AEres	PS-provider	P-ACTIVITY-END response primitive
P-CGreq	PS-provider	P-CONTROL-GIVE request primitive
P-UEreq	PS-provider	P-U-EXCEPTION-REPORT request primitive
P-AIreq	PS-provider	P-ACTIVITY-INTERRUPT request primitive
P-AIres	PS-provider	P-ACTIVITY-INTERRUPT response primitive
P-ADreq	PS-provider	P-ACTIVITY-DISCARD request primitive
P-ADres	PS-provider	P-ACTIVITY-DISCARD response primitive
P-ARreq	PS-provider	P-ACTIVITY-RESUME request primitive

TABLE A-3/X.228 (Part 3 of 3)

Outgoing Event List

Abbreviated name	Source	Name and description
a-ab	RTPM	association aborted, recover
a-res	RTPM	activity resumption by the receiving RTPM
a-ret	RTPM	activity completed, discarded, or interrupted
ass-ab	RTPM	start of association-abort procedure
ass-rec	RTPM	start of association-recovery procedure
ass rec neg	RTPM	association-recovery unsuccessful
next	RTPM	transfer of RTTR APDU
p-ab	RTPM	start of provider-abort procedure
rt-ab	RTPM	RTAB received
tr-discard	RTPM	start of transfer-discard procedure
tr-interr	RTPM	start of transfer-interrupt procedure
tr-p-ab	RTPM	start of procedures transfer-abort followed by provider-abort
tr-pos	RTPM	transfer successful completed
tr-res	RTPM	start of transfer-resumption procedure
transfer	RTPM	start of transfer or transfer-retry procedures
u-exr	RTPM	start of user-exception-report procedure

TABLE A-4/X.228

Predicates

Code	Name and description
p1	RTPM can support the requested application-association
p2	Turn assigned to RTPM
p5	RTPM can support association-recovery
p6	transient rejection of association-recovery
p11	association-initiating RTPM
p30	only one RTTR APDU required to transfer the encoded-APDU-value (no checkpointing)
p31	RTTR APDU is the last one in a series of RTTR APDUs to transfer the encoded-APDU-value
p32	outstanding-minor-syncs < window size
p33	outstanding-minor-syncs = 0
p34	sending RTPM is willing to recover from P-PEind
p35	checkpoint-confirmed (at least on P-MScnf received)
p361	reason parameter value of P-UEind is "receiving ability jeopardized"
p362	reason parameter value of P-UEind is "unrecoverable procedure error"
p363	reason parameter value of P-UEind is "non-specific error"
p364	reason parameter value of P-UEind is "sequence error"
p365	reason parameter value of P-UEind is "local SS-user error"
p41	received RTTR secured
p43	transfer to be resumed was already completed
p44	receiving RTPM is willing to perform and ignore transfer
p45	receiving RTPM can resume the activity
p46	receiving RTPM is willing to perform the association-abort procedure
p91	RTAB abort-reason field value is "user-error"
p92	RTAB abort-reason field value is "permanent-error"
p93	RTAB abort-reason field id value is "transfer-completed"

TABLE A-5/X.228

Specific Action

Code	Name and description
a1	association-initiating RTPM = TRUE
a2	association-initiating RTPM = FALSE
a30	outstanding-minor-syncs = 0, set timer tr to transfer-time, checkpoint-confirmed = FALSE
a31	outstanding-minor-syncs = outstanding-minor-syncs + 1
a32	outstanding-minor-syncs = outstanding-minor-syncs-1, checkpoint-confirmed = TRUE
a33	outstanding-minor-syncs = 0
a35	reset timer tr
a38	set timer rec to local recovery time
a39	reset timer rec
a41	set reason parameter value of P-UEreq to "sequence error"

TABLE A-6/X.228

RTM State Table
Association-establishment

	STA0	STA01	STA02
RT-OPreq	p1: RTORQ [a1] STA01		
RTORQ	p1: RT-OPind [a2] STA02 ¬p1 : RTORJ STA0		
RT-OPres +			p2: RTOAC STA21 ¬p2: RTOAC STA22
RT-OPres-			RTORJ STA0
RTOAC		p2 RT-OPcnf + STA11 ¬p2: RT-OPcnf + STA12	
RTORJ		RT-OPcnf- STA0	
A-ASCcnf-		RT-OPcnf- STA0	
A-PABind		RT-PAind STA0	RT-PAind STA0

TABLE A-7/X.228

RTPM State Table
Association Established, Outside Transfer

	STA11	STA12	STA21	STA22
RT-TRreq	transfer STA30		transfer STA30	
P-ASind		STA40		STA40
P-AIind		P-AIres STA12		P-AIres STA22
P-ARind		[a39] a-res STA45		[a39] a-res STA45
P-ADind		ass-ab STA48		ass-ab STA48
RT-TPreq		RTTP STA12		RTTP STA22
RTTP	RT-TPind STA11		RT-TPind STA21	
RT-TGreq	P-CGreq STA12		P-CGreq STA22	
P-CGind		RT-TGind STA11		RT-TGind STA21
RT-CLreq	A-RELreq STA92			
A-RELind				RT-CLind STA91
A-PABind	ass-rec STA51	ass-rec STA51	ass-rec STA52	ass-rec STA52
RT-UAreq	RTAB STA0	RTAB STA0	RTAB STA0	RTAB STA0
RTAB	rt-ab STA72	rt-ab STA72	rt-ab STA72	rt-ab STA72
rec-timeout		p-ab STA71		p-ab STA71

TABLE A-8/X.228 (Part 1 of 3)

RTPM State Table
Sending RTPM, Transfer

	STA30	STA31	STA32	STA321*
transfer	p30: [a30] P-ASreq RTTR P-AEreq STA32 ¬p30: [a30] P-ASreq next STA30			
next	p32&¬p31: RTTR P-MSreq [a31] next STA30 p32&31: RTTR P-AEreq STA32 ¬p32: STA31			
P-MScnf	[a32] STA30	[a32] next STA30	[a32] STA32	
P-AEcnf			p33: tr-pos STA321	
tr-pos				p11: [a35] RT-TRcnf + STA11 ¬p11: [a35] RT-TRcnf + STA21
tr-timeout	tr-discard [a38] STA35	tr-discard [a38] STA35	tr-discard [a38] STA35	

TABLE A-8/X.228 (Part 2 of 3)

RTPM State Table
Sending RTPM, Transfer

	STA30	STA31	STA32
P-UEind	p361: tr-p-ab STA71 p362: tr-discard STA36 p363: t-discard STA35 p364: tr-discard STA34 p365&p35: tr-interr STA37 p365&¬p35: tr-discard STA36	p361: tr-p-ab STA71 p362: tr-discard STA36 p363: tr-discard STA35 p364: tr-discard STA34 p365&p35: tr-interr STA37 p365&¬p35: tr-discard STA36	p361: tr-p-ab STA71 p362: tr-discard STA36 p363: tr-discard STA35 p364: tr-discard STA34 p365&p35: tr-interr STA37 p365&¬p35: tr-discard STA36
P-PEind	p34&p35: tr-interr STA37 p34&¬p35: tr-discard STA36 ¬p34: tr-p-ab STA71	p34&p35: tr-interr STA37 p34&¬p35: tr-discard STA36 ¬p34: tr-p-ab STA71	p34&p35: tr-interr STA37 p34&¬p35: tr-discard STA36 ¬p34: tr-p-ab STA71

TABLE A-8/X.228 (Part 3 of 3)

RTPM State Table
Sending RTPM, Transfer

	STA30	STA31	STA32
s-problem-1	p35: tr-interr STA37 ¬p35: tr-discard STA36	p35: tr-interr STA37 ¬p35 : tr-discard STA36	p35: tr-interr STA37 ¬p35: tr-discard STA36
s-problem-2	tr-discard STA36	tr-discard STA36	tr-discard STA36
s-problem-3	ass-ab STA38	ass-ab STA38	ass-ab STA38
A-PABind	a-ab STA381	a-ab STA381	a-ab STA381
RT-UAreq	RTAB STA0	RTAB STA0	RTAB STA0
RTAB	rt-ab STA39	rt-ab STA39	rt-ab STA39
RTTP	RT-TPind STA30	RT-TPind STA31	RT-TPind STA32

TABLE A-9/X.228

RTPM State Table
Sending RTPM, Error Handling

	STA34*	STA341	STA35*	STA351	STA36*	STA361
tr-discard	P-ADreq STA341		P-ADreq STA351		P-ADreq STA361	
P-ADcnf		tr-pos STA321		p11: [a35] RT-TRcnf- STA11 ¬p11: [a35] RT-TRcnf- STA21		transfer STA30
A-PABind		a-ab STA381		a-ab STA381		a-ab STA381
RT-UAreq		RTAB STA0		RTAB STA0		RTAB STA0
RTAB		rt-ab STA39		rt-ab STA39		rt-ab STA39
RTTP		RT-TPind STA341		RT-TPind STA351		RT-TPind STA361
tr-timeout		[a38] STA351		[a38] STA351		[a38] STA351
rec-timeout				tr-p-ab STA71		

TABLE A-10/X.228

RTPM State Table
Sending RTPM, Error Handling

	STA37*	STA371	STA372*
tr-interr	P-AIreq STA371		
P-AIcnf		tr-res STA372	
tr-res			p35: [a33] P-ARreq next STA30 ¬p35: P-ARreq tr-discard STA36
A-PABind		a-ab STA381	
RT-UAreq		RTAB STA0	
RTAB		rt-ab STA39	
RTTP		RT-TPind STA371	
tr-timeout		tr-p-ab STA71	

TABLE A-11/X.228

RTPM State Table
Sending RTPM, Error Handling

	STA38*	STA381*	STA39*
ass-ab	RTAB a-ab STA381		
a-ab		p11: ass-rec STA53 ¬p11: STA532	
rt-ab			p93&p11: RT-TRcnf + ass-rec STA51 p93&¬p11 RT-TRcnf + ass-rec STA52 p91: RT-TRcnf - RT-UAind STA0 p92: RT-TRcnf - RT-PAind STA0 ¬p91&¬p92: a-ab STA381

TABLE A-12/X.228

RTPM State Table
Receiving RTPM

	STA40	STA41	STA400	STA410	STA42
RTTR	STA41		STA410		
P-MSind		p41: P-MSres STA40		P-MSres STA400	
P-AEind		RT-TRind P-AEres a-ret STA43		P-AEres a-ret STA43	
P-AIind	[a38] P-AIres a-ret STA43	[a38] P-AIres a-ret STA43	[a38] P-AIres a-ret STA43	[a38] P-AIres a-ret STA43	P-AIres a-ret STA43
P-ADind	P-ADres a-ret STA43	P-ADres a-ret STA43	P-ADres a-ret STA43	P-ADres a-ret STA43	[a39] P-ADres a-ret STA43
P-PEind	STA40	STA41	STA400	STA410	STA42
r-problem-1	u-exr STA44	u-exr STA44	u-exr STA44	u-exr STA44	
r-problem-2	ass-ab STA48	ass-ab STA48	ass-ab STA48	ass-ab STA48	ass-ab STA48
A-PABind	a-ab STA481	a-ab STA481	a-ab STA481	a-ab STA481	a-ab STA481
RT-TPreq	RTTP STA40	RTTP STA41	RTTP STA400	RTTP STA410	
RT-UAreq	RTAB STA0	RTAB STA0	RTAB STA0	RTAB STA0	RTAB STA0
RTAB	rt-ab STA49	rt-ab STA49	rt-ab STA49	rt-ab STA49	rt-ab STA49
rec-timeout					RT-PAind RTAB STA0

TABLE A-13/X.228 (Part 1 of 2)

RTPM State Table
Receiving RTPM Error Handling

	STA43*	STA44*	STA45*
a-ret	p11: STA12 ¬p11: STA22		
u-exr		P-UEreq: [a38] STA42	
a-res			¬p43&p45: STA40 p43&p44&p45: p43&¬p44&p45: [a41] u-exr STA44 ¬p45&¬p46: u-exr STA44 ¬p45&p46: ass-ab STA48

TABLE A-13/X.228 (Part 2 of 2)

RTPM State Table
Receiving RTPM Error Handling

	STA48*	STA481*	STA49*
ass-ab	RTAB a-ab STA481		
a-ab		p11: ass-rec STA54 ¬p11: ass-rec STA542	
rt-ab			p91: RT-UAind STA0 p92: RT-PAind STA0 ¬p91&¬p92: a-ab STA481

TABLE A-14/X.228 (Part 1 of 2)

RTPM State Table
Association-recovery Outside Transfer

	STA51*	STA510	STA52
ass-rec	p5: [a38] RTORQ STA510 ¬p5: p-ab STA70		[a38] STA52
RTORQ			p5&p2: [a39] RTOAC STA21 p5&¬p2: [a39] RTOAC STA22 ¬p5&p6: RTORJ STA52 ¬p5&¬p6: RTORJ p-ab STA70
RTOAC		p5&p2: [a39] STA11 p5&¬p2: [a39] STA12	

TABLE A-14/X.228 (Part 2 of 2)

RTPM State Table
Association-recovery Outside Transfer

	STA51*	STA510	STA52
RTORJ		ass-rec-neg STA51	
A-ASCcnf-		ass-rec-neg STA51	
A-PABind		ass-rec-neg STA51	
ass-rec-neg	p6: ass-rec STA51 ¬p6: p-ab STA70		
rec-timeout		p-ab STA71	p-ab STA70

TABLE A-15/X.228

RTPM State Table
Association-recovery During Transfer

	STA53*	STA531	STA532	STA54*	STA541	STA542
ass-rec	RTORQ STA531			[a38] RTORQ STA541		[a38] STA542
RTORQ			p5&p2: RTOAC tr-res STA372 ¬p5&p6: RTORJ STA532 ¬p5&¬p6: RTORJ tr-p-ab STA70			p5&¬p2: RTOAC [a39] STA22 ¬p5&p6: RTORJ STA542 ¬p5&¬p6: RTORJ p-ab STA70
RTOAC		tr-res STA372			[a39] STA12	
RTORJ		ass-rec-neg STA53			ass-rec-neg STA54	
A-ASCcnf-		ass-rec-neg STA53			ass-rec-neg STA54	
A-PABind		ass-rec-neg STA53			ass-rec-neg STA54	
ass-rec-neg	p6: ass-rec STA53 ¬p6: tr-p-ab STA70			p6: ass-rec STA54 ¬p6: p-ab STA70		
tr-timeout		tr-p-ab STA71	tr-p-ab STA70			
rec-timeout					p-ab STA71	p-ab STA70

TABLE A-16/X.228

RTPM State Table
Abort and Association-release

	STA70*	STA71*	STA72*	STA91	STA92
tr-p-ab	RT-TRcnf- RT-PAind STA0	RT-TRcnf- RTAB RT-PAind STA0			
p-ab	RT-PAind STA0	RT-PAind RTAB STA0			
rt-ab			p91: RT-UAind STA0 p92: RT-PAind STA0		
RT-CLres				A-RELres STA0	
A-RELcnf					RT-CLcnf STA0
A-PABind					p-ab STA70
RTAB					rt-ab STA72
RT-UAreq				RTAB STA0	

ANNEX B

(to Recommendation X.228)

Differences between this Recommendation and Recommendation X.410-1984

This annex is not part of this Recommendation.

This annex describes the technical differences between the protocol for Reliable Transfer of this Recommendation and the corresponding protocol of CCITT Recommendation X.410-1984.

In X.410-1984 mode this Recommendation and its use of ACSE and Presentation service is bit compatible to Recommendation X.410-1984 under consideration of the clarifications and errata of the X.400-series Implementors Guide V.5.

B.1 *Application protocol data units*

B.1.1 *PConnect*

1) The Set type and its two elements (Data Transfer Syntax and pUser Data) are now Presentation protocol control information (PPCI). The Elements of the RTORQapdu are the elements of the SETpUserData.
2) The application Protocol element is now OPTIONAL and solely used in X.41-1984 mode.
3) Implicit tagging of the SET in normal mode.

B.1.2 *PAccept*

1) The SET type and its two elements (DataTransferSyntax and pUserData) are now Presentation protocol control information. The elements of the RTOACapdu are the elements of the SET pUserData.
2) Implicit tagging of the SET in normal mode.

B.1.3 *PRefuse*

1) The SET type is now Presentation protocol control information. The elements of the RTORJapdu are the elements of the PRefuse SET.
2) Implicit tagging of the SET in normal mode.
3) Additional optional user data field in normal mode.

B.1.4 *Datatransfersyntax*

This information is now Presentation protocol control information.

B.1.5 *AbortInformation*

1) The SET type is now Presentation protocol control information. The elements of the RTABapdu are the elements of the AbortInformation SET.
2) Implicit tagging of the SET in normal mode.
3) Additional optional user data field in normal mode.

B.1.6 *AbortReason*

Add: Values (5) to (6) inclusive. Value (7) was added by the Addendum of the X.400 series Implementors Guide Version 5.

B.2 *Procedures and Mapping*

General Mapping onto used services

Change: From: onto Session Services

To: Mapping onto ACSE and Presentation services.

ANNEX C

(to Recommendation X.228)

Summary of Assigned Object Identifier Values

This annex is not an integral Part of this Recommendation.

This annex summarizes the object identifier values assigned in Recommendations X.218 and X.228.

{ **joint-iso-ccitt reliable-transfer (3) apdus (0)** }	- - *ASN. 1 module defined in X.228*
{ **joint-iso-ccitt reliable-transfer (3) aseID (1)** }	- - *RTSE identifier* - - *defined in X.228*
{ **joint-iso-ccitt transfert-fiable (3) abstract-syntax (2)** }	- - *Abstract syntax name* - - *defined in X.228*